CSE 332: Data Abstractions

Section 2: Heaps, Asymptotics, & Recurrences

0. Heaps

Insert 1, 3, 2, 4, 6, 8, 7, 5 into a *min heap*.
Now, insert the same values into a *max heap*.
Now, insert the same values into a *min heap*, but use Floyd's `buildHeap` algorithm.

1. Big-Oh Proofs

For each of the following, prove that $f \in \mathcal{O}(g)$.

(a) $f(n) = 7n$ $g(n) = \frac{n}{10}$

(b) $f(n) = 1000$ $g(n) = 3n^3$

(c) $f(n) = 7n^2 + 3n$ $g(n) = n^4$

(d) $f(n) = n + 2n \lg n$ $g(n) = n \lg n$

2. Is Your Program Running? Better Catch It!

For each of the following, determine the asymptotic worst-case runtime in terms of n.

(a)

```
1 int x = 0;
2 for (int i = n; i >= 0; i--) {
3    if ((i % 3) == 0) {
4       break;
5    }
6    else {
7       x += n;
8    }
9 }
```

(b)

```
1 int x = 0;
2 for (int i = 0; i < n; i++) {
3    for (int j = 0; j < (n * n / 3); j++) {
4       x += j;
5    }
6 }
```

(c)

```
1 int x = 0;
2 for (int i = 0; i <= n; i++) {
3    for (int j = 0; j < (i * i); j++) {
4       x += j;
5    }
6 }
```

3. Induction Shminduction

Prove $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$ by induction on n.

4. The Implications of Asymptotics

For each of the following, determine if the statement is true or false.

(a) $f(n) \in \Theta((g(n)) \rightarrow f(n) \in \mathcal{O}(g(n))$

(b) $f(n) \in \Theta(g(n)) \rightarrow g(n) \in \Theta(f(n))$

(c) $f(n) \in \Omega((g(n) \rightarrow g(n) \in \mathcal{O}(f(n))$

5. Asymptotic Analysis

For each of the following, determine if $f \in \mathcal{O}(g)$, $f \in \Omega(g)$, $f \in \Theta(g)$, several of these, or none of these.

(a) $f(n) = \log n$ $g(n) = \log \log n$

(b) $f(n) = 2^n$ $g(n) = 3^n$

(c) $f(n) = 2^{2n}$ $g(n) = 2^n$

6. Recurrences and Closed Forms

For each of the following code snippets, find a recurrence for the worst case runtime of the function, and then find a closed form for the recurrence.

(a) Consider the function f:

```
1 f(n) {
2    if (n == 0) {
3       return 1;
4    }
5    return 2 * f(n - 1) + 1;
6 }
```

- Find a recurrence for $f(n)$.

- Find a closed form for $f(n)$.

7. Big-Oh Bounds for Recurrences

For each of the following, find a Big-Oh bound for the provided recurrence.

(a) $T(n) = \begin{cases} 1 & \texttt{if } n = 0 \\ T(n-1) + 3 & \texttt{otherwise} \end{cases}$

(b) $T(n) = \begin{cases} 1 & \texttt{if } n = 0 \\ T(n-1) + T(n-2) + 3 & \texttt{otherwise} \end{cases}$