CSE 332 - Section 2 Worksheet

1. Big-Oh Proofs

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

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

b) $f(n) = 1000$ $\quad\quad$ $g(n) = 3n^2$

c) $f(n) = 2^n$ $\quad\quad$ $g(n) = 3^{2n}$

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

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

2. Big-Theta Proofs

For each of the following, prove that $f(n) \in \Theta(g)$:

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

b) $f(n) = n^3 + 10n$ $g(n) = 3n^3$

CSE 332 - Section 2 Worksheet

3. Algorithm Running Time

Consider the following method which finds the number of unique Strings within a given array of length n.

```
1  int numUnique(String[] values) {
2      boolean[] visited = new boolean[values.length]
3      for (int i = 0; i < values.length; i++) {
4          visited[i] = false
5      }
6      int out = 0
7      for (int i = 0; i < values.length; i++) {
8          if (!visited[i]) {
9              out += 1
10             for (int j = i; j < values.length; j++) {
11                 if (values[i].equals(values[j])) {
12                     visited[j] = true
13                 }
14             }
15         }
16     }
17     return out;
18 }
```

Determine a $\Theta(\cdot)$ bound on each function below. Start by (1) constructing an equation that models each function then (2) simplifying and finding a closed form.

a) $f(n)$ = the worst-case runtime of `numUnique`

b) $g(n)$ = the best-case runtime of `numUnique`