1 Big O

```
Big O Additional Problems:
```

- 1.1 O(b)
- 1.2 O(b)
- 1.3 O(1)
- 1.4 $O(\frac{a}{b})$
- $1.5 \operatorname{O}(\log_2(n))$
- 1.6 O($\sqrt[2]{n}$)
- $1.7~\mathrm{O}(n)$ in the case that each node has 1 child in the same direction (degenerate tree).
 - 1.8 O(n), as you have no heuristics on where the node is located
 - 1.9 $O(n^2)$ as each copy is $1 + 2 + 3 + ... + n 1 \le n(n) \in O(n^2)$
- $1.10 \text{ O}(\log_1 0(n))$, which is equalizent to $\text{O}(\log_2(n))$ (up to a constant factor for change of base)
- 1.11 Checking if is in order takes O(s) in size of string s, otherwise makes successive calls to every possible string with c^s possibilities, so $O(s*c^s)$
- 1.12 Total is $O(b \log b)$ for mergesort $+ a \log b$ for binary searching b for each int in a. So, $O((a+b) \log b)$.

2 Arrays & Strings

Chapter 1: Arrays & Strings Interview Questions 1.1

1.2