CSI 2110 A1

a) $n\log_2(n^3) + \sqrt{n}$ is $O(n\log_{10}n)$

Notice that $\sqrt{n} < n \leq n\log_2 n$ for all $n \geq 2$

$\therefore n\log_2(n^3) + \sqrt{n}$

$\leq 3n\log_2 n = 3n \cdot \frac{\log_{10} n}{\log_{10} 2} = \frac{3n}{\log_{10} 2} \cdot \log n$ for all $n \geq 2$

$\therefore n\log_2(n^3) + \sqrt{n}$ is $O(n\log_{10} n)$ True

$n_0 = 2$, $c = \frac{3}{\log_{10} 2}$

b) $3^{n-1} + n^3$ is $\Omega(2^n)$

$3^{n-1} + n^3 = \frac{1}{3}3^n + n^3 \geq \frac{1}{3}3^n \geq \frac{1}{3}2^n$ for all $n \geq 0$

$\therefore n_0 = 0$, $c = \frac{1}{3}$ $3^{n-1} + n^3$ is $\Omega(2^n)$ True

c) $n^3\log_{10} n + 10n + 100$ is $\Theta(n^4)$

False $\because$ it is True Fcns is $O(n^4)$ and $\Omega(n^4)$

but for $n^3\log_{10} n$ we notice that $\log_{10} n < n$.

$\therefore$ Fcns cannot be $\Omega(n^4)$

Thus, we proved that is False.

2. a) if the pretixes are sorted from smallest to largest, the code will never reach the if (s.charAt(i-1) > s.charAt(i)). And that is the worst-case. T(n) is O(n²)

b) if the pretixes are sorted from largest to smallest. since the second pretix is bigger than the first one. the java code will break. B(n) is O(1)

3. a) for queue: first in, first out.

while (!D.isEmpty()) { Q.enqueue(D.removeFirst()); }

while (!Q.isEmpty()) { D.insertLast(Q.dequeue()); }

b) for stack: last in, first out.

while (!D.isEmpty()) { S.push(D.removeLast()); }

while (!S.isEmpty()) { D.insertLast(S.pop()); }

4.

```
Stack<Integer> s = new Stack<Integer>();
int index = 1;
Iterator<Integer> iterator = L.iterator();
while (iterator.hasNext()) {
    a = iterator.next();
    if (L.size() % 2 == 0) {
        if (index <= L.size()/2) {
            s.push(a);
        } else {
            if (s.pop() + a != 10) { return false; }
        }
    } else {
        if (index < Math.floor(L.size()/2)) {
            s.push(a);
        } else if (index == Math.floor(L.size()/2)) {
            if (a != 5) { return false; }
        } else { if (s.pop() + a != 10) { return false; } }
    }
return true;
```