Question 1:

Use the definitions of O and of Θ in order to show the following:

a. $5n^3 + 2n^2 + 3n = O(n^3)$

b. $\sqrt{7n^2 + 2n - 8} = \Theta(n)$

c. Show that if *d(n)=O(f(n))* and *e(n)=O(g(n))*, then the product *d(n)e(n)* is O*(f(n)g(n))*.

PART A

$5n^3 + 2n^2 + 3n = O(n^3)$

Proof:

Let $f(n) = 5n^3 + 2n^2 + 3n$

$g(n) = n^3$

By definition of O notation, $f(n) \le c\,g(n)$ for any $n > n_0$

$5n^3 + 2n^2 + 3n \le 5n^3 + 2n^3 + 3n^3$

$-5n^3 + 2n^2 + 3n \le 0$

$n(5n^2 - 2n - 3) \ge 0$

$(n \ge 0) \vee (5n^2 - 2n - 3 \ge 0)$

$\Delta = 4 + 60 = 64$

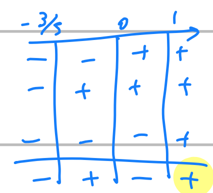
$n_{1/2} = \frac{2 \pm 8}{10} = <^{-3/5}_{1}$

$(5n+3)(n-1) \ge 0$

$n \ge 0$

$n \ge -3/5$

$n \ge 1$



$n(5n+3)(n-1) \ge 0$

$\Rightarrow n \ge 1 \Rightarrow n_0 = 1$

$5n^3 + 2n^3 + 3n^3 = 10n^3$

$\therefore 5n^3 + 2n^2 + 3n \le 10n^3$

$\therefore C = 10$

$\therefore 5n^3 + 2n^2 + 3n = O(n^3)$ QED

PART B

$$\sqrt{7n^2+2n-8} = \theta(n)$$

Proof:

Let $f(x) = \sqrt{7n^2+2n-8}$

Factor an n^2 from $f(x)$

$$f(x) = n\sqrt{7+\frac{2}{n}-\frac{8}{n^2}}$$

For $n \geq 1$ (UPPER)

$$7+\frac{2}{n}-\frac{8}{n^2} \leq 7+\frac{2}{n} \leq 7+\frac{2}{1} = 9$$

$$\Rightarrow \sqrt{9} = 3$$

For $n \geq 1$ (LOWER)

$$7+\frac{2}{n}-\frac{8}{n^2} \geq 7+\frac{2}{n}-\frac{8}{n} = 7-\frac{6}{n} \leq 7-\frac{6}{1} = 7-6=1$$

$$\Rightarrow \sqrt{1} = 1$$

$$1 \leq \sqrt{7+\frac{2}{n}-\frac{8}{n^2}} \leq 3$$

$$n \leq n\sqrt{7+\frac{2}{n}-\frac{8}{n^2}} \leq 3n$$ because n is a positive integer

Let $g(x) = n$

$c_1 g(n) \leq f(n) \leq c_2 g(n)$ for any $n > n_0 \Rightarrow n_0 > 1$

$c_1 = 1$

$c_2 = 3$

$$1 \cdot n \leq n\sqrt{7+\frac{2}{n}-\frac{8}{n^2}} \leq 3 \cdot n$$

$$\therefore \sqrt{7n^2+2n-8} = \theta n \quad \text{QED}$$

PART C

$$\left(d(n) = O(f(n))\right) \wedge \left(e(n) = O(g(n))\right) \rightarrow \left(d(n)e(n) = O(f(n)\,g(n))\right)$$

Proof:

Let $d(n) = O(f(n))$ and $e(n) = O(g(n))$

By definition of $O(n)$, $d(n) \leq c_1 f(n)$ for all $n \geq n_1$

$e(n) \leq c_2 g(n)$ for all $n \geq n_2$

By algebra, $d(n) \cdot e(n) \leq c_1 f(n) c_2 g(n)$ for all $n \geq \left(\frac{n_1 + n_2 + |n_1 - n_2|}{2}\right)$

By algebra, $d(n) \cdot e(n) \leq c_1 \cdot c_2 \; f(n) g(n)$

In other words the maximum between n_1 and n_2

Let $c_3 = c_1 \cdot c_2$

By substitution, $d(n) \cdot e(n) \leq c_3 f(n) g(n)$

$\therefore$ By definition of $O(n)$, $d(n) \cdot e(n) = O(f(n) g(n))$ QED

Question 2:

Give a θ characterization, in terms of *n*, of the running time of the following four functions:

```
def example1(lst):
 """Return the sum of the prefix sums of sequence S."""
θ(1)  [ n = len(lst)
      [ total = 0
θ(n²) [ for j in range(n):
      [     θ(j) [ for k in range(1+j):
      [     θ(n)*[     θ(1) [ total += lst[k]
θ(1)  [ return total
```

* in the worst case it would be θ(n)

$T_{worst}(n) = \theta(n^2)$

```
def example2(lst):
"""Return the sum of the prefix sums of sequence S."""
      [ n = len(lst)
θ(1)  [ prefix = 0
      [ total = 0
      [ for j in range(n):
θ(n)  [     θ(1) [ prefix += lst[j]
      [          [ total += prefix
θ(1)  [ return total
```

$T_{worst}(n) = \theta(n)$

```
def example3(n):
θ(1)         [ i = 1
             [ sum = 0
θ(log₂(n))   [ while (i < n*n):
             [     θ(1) [ i *= 2
             [          [ sum += i
θ(1)         [ return sum
```

$T_{worst}(n) = \theta(\log_2(n))$

```
def example4(n):
θ(1)  [ i = n
      [ sum = 0
      [ while (i > 1):
θ(n)  [     θ(i) [ for j in range(i):
      [     θ(n)*[     θ(1) [ sum += i*j
      [     θ(1) [ i //= 2
θ(1)  [ return sum
```

* in the worst case it would be θ(n)

$T_{worst}(n) = \theta(n)$