1. Let $T(n) = \frac{1}{2}n^2 + 3n$. Prove the following.

(a). $T(n)$ is not $O(n)$ [Hint: proof by contradiction as Proposition 2.2 in slides]

Proof. Suppose $T(n) = O(n)$. Then for c, $n_0 \in \mathbb{N}$, $T(n) \leq c \cdot n$ for all $n \geq n_0$.

$$\begin{aligned}\frac{1}{2}n^2 + 3n &\leq c \cdot n \\ \frac{1}{2}n^2 &\leq c \cdot n - 3n \\ &\leq (c-3)n \\ n^2 &\leq 2(c-3)n \\ n &\leq 2(c-3)\end{aligned}$$

There is no c which can satisfy this inequality. Specifically, for any c we may choose, we may always find a sufficiently large enough n such that the inequality is dissatisfied. Thus, $T(n)$ is not $O(n)$. □

(b). $T(n) = \Omega(n)$ [Hint: find c and n_0 to satisfy the inequality]

Proof. Let $T(n) = \Omega(n)$. Then, there must exist some c, $n_0 \in \mathbb{N}$ such that

$$T(n) \geq c \cdot n$$

for all $n \geq n_0$.

$$\frac{1}{2}n^2 + 3n \geq c \cdot n$$

Suppose $c = 1$, and $n_0 = 1$.

$$\begin{aligned}\frac{1}{2}n^2 + 3n &\geq n \\ \frac{1}{2}n + 3 &\geq 1\end{aligned}$$

We can see that for all $n \geq 1$, $T(n) \geq n$. Thus, $T(n) = \Omega(n)$. □

(c). $T(n) = \Theta(n^2)$ [Hint: find c_1, c_2, and n_0 to satisfy the two inequalities]

Proof. Let $T(n) = \Theta(n^2)$. Then we have two inequalities which must be satisfied. Let $c_1, c_2, n_0 \in \mathbb{N}$, then

$$T(n) \leq c_1 \cdot n^2 \tag{1}$$

$$T(n) \geq c_2 \cdot n^2 \tag{2}$$

for all $n \geq n_0$, respectively.

For (1), we may choose $c_1 = 4$, and $n_0 = 1$, giving us

$$\frac{1}{2}n^2 + 3n \leq 4n^2$$
$$\frac{1}{2}n + 3 \leq 4n$$

which holds for all $n \geq 1$.

For (2), we may choose $c_2 = \frac{1}{2}$ and $n_0 = 1$, giving us

$$\frac{1}{2}n^2 + 3n \geq \frac{1}{2}n^2$$
$$\frac{1}{2}n + 3 \geq \frac{1}{2}n$$

which holds for all $n \geq 1$.

We have found sufficient c_1, c_2, n_0 satisfying the definition of $\Theta(n^2)$. Thus, $T(n) = \Theta(n^2)$. □

(d). $T(n) = O(n^3)$ [Hint: find c and n_0 to satisfy the inequality]

Proof. Let $T(n) = O(n^3)$. Let c, $n_0 \in \mathbb{N}$. Then we have

$$T(n) \leq c \cdot n^3$$

for all $n \geq n_0$. Let $c = 10$, $n_0 = 1$. Then

$$\frac{1}{2}n^2 + 3n \leq 10n^3$$
$$\frac{1}{2}n + 3 \leq 10n^2$$

for all $n \geq 1$, satisfying the definition of $O(n^3)$. Thus, $T(n) = O(n^3)$. □

2. Let $T_1(n) = O(f(n))$ and $T_2(n) = O(g(n))$.

(a). Prove that $T_1(n) + T_2(n) = O(f(n) + g(n))$.

Proof. Let $T_1(n) = O(f(n))$, $T_2(n) = O(g(n))$. Let c_1, c_2, n_1, $n_2 \in \mathbb{N}$. Then the following inequalties must be satisfied.

$$T_1(n) \leq c_1 \cdot f(n) \tag{3}$$

$$T_2(n) \leq c_2 \cdot g(n) \tag{4}$$

for $n \geq n_1$, $n \geq n_2$ respectively.

Adding (3), (4), we get

$$T_1(n) + T_2(n) \leq c_1 \cdot f(n) + c_2 \cdot g(n)$$

Suppose $c_3 = c_1 + c_2$, and suppose $n_3 = \max(n_1, n_2)$. Then the following inequality also holds.

$$\begin{aligned} T_1(n) + T_2(n) &\leq c_3 \cdot f(n) + c_3 \cdot g(n) \\ &\leq c_3 \cdot (f(n) + g(n)) \end{aligned}$$

for all $n \geq n_3$.

This precisely satisfies the definition of $O(f(n)+g(n))$. Therefore, $T_1(n)+T_2(n) = O(f(n) + g(n))$. $\square$

(b). Prove that $T_1(n) \cdot T_2(n) = O(f(n) \cdot g(n))$.

Proof. Let $T_1(n) = O(f(n))$, $T_2(n) = O(g(n))$. Let c_1, c_2, n_1, $n_2 \in \mathbb{N}$. Then the following inequalties must be satisfied.

$$T_1(n) \leq c_1 \cdot f(n) \tag{5}$$

$$T_2(n) \leq c_2 \cdot g(n) \tag{6}$$

for $n \geq n_1$, $n \geq n_2$ respectively.

Multiplying (5), (6), we get

$$T_1(n) \cdot T_2(n) \leq c_1 \cdot f(n) \cdot c_2 \cdot g(n)$$

Let $c_3 = c_1 \cdot c_2$, and let $n_3 = n_1 \cdot n_2$. Then we have

$$T_1(n) \cdot T_2(n) \leq c_3 \cdot (f(n) \cdot g(n))$$

holding for all $n \geq n_3$, satisfying the definition of $O(f(n) \cdot g(n))$. Therefore, $T_1(n) \cdot T_2(n) = O(f(n) \cdot g(n))$. $\square$

3. Let f and g be non-decreasing real-valued functions defined on the positive integers, with $f(n)$ and $g(n)$ at least 2 for all $n \geq 1$. Assume that $f(n) = O(g(n))$, and let c be a positive constant.

Is $f(n) \cdot log_2(f(n)^c) = O(g(n) \cdot log_2(g(n)))$? Write your argument.

Ans: Assuming f and g are monotonic, we maintain that $f(n)$, $g(n) \geq 2$ for all $n \geq 1$.

Let $f(n) = O(g(n))$, and let c, $c_1 \in \mathbb{R}$, c, $c_1 > 0$, and let $n_0 \in \mathbb{N}$. Then we have the following.

$$f(n) \leq c_1 \cdot g(n) \tag{7}$$

for all $n \geq n_0$. Let $n_1 = log_2(n_0^c)$. Then

$$\begin{aligned} log_2(f(n)^c) &\leq log_2((c_1 \cdot g(n))^c) \\ &\leq c \cdot log_2(c_1 \cdot g(n)) \end{aligned}$$

for all $n > n_1$, then we have that $log_2(f(n)^c) = O(log_2(g(n)))$. From our previous result for 2(a), we know that the multiplication of functions has a big O of the product of their composite big Os. Thus

$$f(n) \cdot log_2(f(n)^c) = O(g(n) \cdot log_2(g(n)))$$

4. Show that $a^{log_b(n)} = n^{log_b(a)}$. (Hint: To verify the equality, take the logarithm base-b of both sides.)

Ans:

$$\begin{aligned} a^{log_b(n)} &= n^{log_b(a)} \\ log_b(a^{log_b(n)}) &= log_b(n^{log_b(a)}) \\ log_b(n) \cdot log_b(a) &= log_b(a) \cdot log_b(n) \end{aligned}$$

5. Order the following functions by growth rate:

(a). $2^{log_2(n)}$

(b). $2^{2^{log_2(n)}}$

(c). $n^{\frac{5}{2}}$

(d). 2^{n^2}

(e). $n^2 \cdot log_2(n)$

(Note that exponentiation base-2 and logarithm base-2 are inverse operations, so "transform" (a) and (b) into something else by getting rid of log.)

Ans: (a) $= n$, (b) $= 2^n$.

Therefore

$n < n^2 \cdot log_2(n) < n^{\frac{5}{2}} < 2^n < 2^{n^2}$

6. Consider the Abstract Data Type (ADT) `Polynomial`-in a single variable x—whose operations include the following:

- `degree()`: returns the degree of a polynomial
- `coefficient(power)`: returns the coefficient of the x^{power} term
- `changeCoefficient(newCoefficient, power)`: replaces the coefficient of the x^{power} term with newCoefficient

We consider only polynomials whose exponents are nonnegative integers. For instance, $\texttt{p} = 4x^5 + 7x^3 - x^2 + 9$. The following examples demonstrate the ADT operations on polynomial object `p`.

- `p.degree()` is 5 (the highest power of a term with a nonzero coefficient)
- `p.coefficient(3)` is 7 (the coefficient of the x^3 term)
- `p.coefficient(4)` is 0 (the coefficient of a missing term is implicitly 0)
- `p.changeCoefficient(-3, 7)` produces the polynomial $-3x^7 + 4x^5 + 7x^3 - x^2 + 9$.

Using these ADT operations, write statements in C++ syntax to perform the following tasks:

(a). Given polynomial object `p`, display the coefficient of the term that has the highest power.

Ans:

```
cout << p.coefficient(p.power()) << '\n';
```

(b). Given polynomial object `p`, increase the coefficient of the x^3 term by 8.

Ans:

```
p.changeCoefficient(p.coefficient(3)+8,3);
```

(c). Compute polynomial object `r` to be the sum of two polynomial objects `p` and `q`.

Ans:

```
Polynomial r;
for(int i = 0; i <= max(p.degree(), q.degree()); ++i){
  r.changeCoefficient(p.coefficient(i)+q.coefficient(i), i);
}
```