Li Liu

1. Order the functions in increasing order:

10^5	10^5 is a constant, so it is the slowest.
$\lg\lg n$	$\lg\lg n$
$\lg(n^5)$	$\lg(n^5) = 5\lg n$
$n^{\frac{1}{4}}$	$n^{\frac{1}{4}}$, $\frac{1}{4} < \frac{1}{2}$
$\sqrt{n}$	$\sqrt{n} = n^{\frac{1}{2}}$, $n^{\frac{1}{2}} < n^1$
$10n$ $2^{\lg n}$ same growth rate	$2^{\lg n} = n$. $10n$ linear function
$n\lg n$	
$n^3 + n^2 + n\lg n + 2^{10}$	n^3 polynomial function
$2^{\frac{n}{5}}$	$2^{\frac{n}{5}} = (2^{\frac{1}{5}})^n = (\sqrt[5]{2})^n$ exponential function.

$\lg(n^5)$ and $n^{\frac{1}{4}}$:

$\lg(\lg(n^5)) = \lg(5\lg n) = \lg 5 + \lg\lg n$

$\lg(n^{\frac{1}{4}}) = \frac{1}{4}\lg n$. $\lg\lg n$ grows slower than $\lg n$.

Thus $\lg(n^5)$ grows slower than $n^{\frac{1}{4}}$.

$n\lg n$ and n^3:

$n\lg n$ grows slower than n^2

n^2 grows slower than n^3.

---

n^3 and $2^{\frac{n}{5}}$:

polynomial functions grow slower than exponential function.

$\log(n^3) = 3\log n$, $\log(2^{\frac{n}{5}}) = \log((\sqrt[5]{2})^n) = n\log\sqrt[5]{2} = \frac{1}{5}n$.

$\log n$ grows slower than n, so $\log(n^3)$ slower than $2^{\frac{n}{5}}$.

2. (a). Claim. $f(n) = n^2 + 2n + 5 \in \Theta(n^2)$.

Proof. First, we will show that $f(n) = O(n^2)$.

Select $C = 8$, $N = 1$. We will show that $f(n) \le 8n^2$.

For $n \in \mathbb{N}$, $n \ge 1$, $n^2 \ge 1$. and $n^2 \ge n$.

Thus $f(n) = n^2 + 2n + 5 \le n^2 + 2n^2 + 5n^2$

$\le 8n^2$

Therefore, for $n > 1$, $f(n) \le 8n^2$, $f(n) = O(n^2)$.

Second, we will show that $f(n) = \Omega(n^2)$.

Select $C = 1$ and $N = 1$. We will show that for any $n \ge 1$, $f(n) \ge g(n)$.

Since $f(n) = n^2 + 2n + 5$, $g(n) = n^2$.

$n \ge 1$. we have $2n + 5 \ge 2 \cdot 1 + 5 = 7 > 0$.

Adding n^2 on both sides of the inequality, we have.

$n^2 + 2n + 5 \ge n^2$.

Therefore $f(n) = \Omega(n^2)$.

$f(n) = O(n^2)$ and $f(n) = \Omega(n^2)$. so $f(n) = \Theta(n^2)$.

The claim follows. 囧

(b). Claim: $f(n)=\sum_{i=1}^{n}((i-1)\times i)\in\Theta(n^3)$.

proof. $f(n)=\sum_{i=1}^{n}((i-1)\times i)=\sum_{i=1}^{n}(i^2-i)$.

$=\sum_{i=1}^{n}i^2-\sum_{i=1}^{n}i$.

Since $\sum_{i=1}^{n}i^2=\frac{n(n+1)(2n+1)}{6}$ and $\sum_{i=1}^{n}i=\frac{n(n+1)}{2}$

Thus $f(n)=\frac{n(n+1)(2n+1)}{6}-\frac{n(n+1)}{2}$

$=\frac{n(n+1)(2n+1)}{6}-\frac{3n(n+1)}{6}$

$=\frac{n(n+1)(2n+1)-3n(n+1)}{6}$

$=\frac{(n+1)(2n^2+n-3n)}{6}$

$=\frac{2(n+1)n(n-1)}{6}=\frac{n(n-1)(n+1)}{3}=\frac{n^3-n}{3}$

First, we will show that $f(n)=O(n^3)$.

Let $C=\frac{1}{3}$, $N=1$, we will prove that for any $n\geq 1$ $f(n)\leq\frac{1}{3}n^3$

For $n\geq 1$, multiplying $-\frac{1}{3}$ on both sides of the inequality,

we have $-\frac{1}{3}n\leq-\frac{1}{3}$, thus $-\frac{1}{3}n<0$.

Adding $\frac{1}{3}n^3$ on both sides of $-\frac{1}{3}n<0$, we get

$\frac{1}{3}n^3-\frac{1}{3}n<\frac{1}{3}n^3$.

Therefore $f(n)=\frac{n^3-n}{3}\leq\frac{1}{3}n^3=\frac{1}{3}g(n)$, $f(n)=O(n^3)$.

Second, we will show that $f(n)=\Omega(n^3)$.

Let $C=\frac{1}{6}$, $N=2$, we will show that $f(n)\geq\frac{1}{6}n^3$.

Since $n\geq 2$, $n^2\geq 4$, $n^2-2\geq 4-2=2\geq 0$

Multiplying the inequalities $n^2-2\geq 0$ and $n>0$ gives that

$n(n^2-2) \geq 0$,

$n^3 - 2n \geq 0$

$n^3 \geq 2n$

Multiplying $\frac{1}{6}$ on both sides of the inequality, we have

$\frac{1}{6}n^3 \geq \frac{1}{3}n$, $\frac{1}{6}n^3 - \frac{1}{3}n \geq 0$.

Since $n^2 \geq 0$, adding $\frac{1}{6}n^3$ on both sides of the inequality, we have $\frac{1}{6}n^3 - \frac{1}{3}n + \frac{1}{6}n^3 \geq \frac{1}{6}n^3$.

$\frac{1}{3}n^3 - \frac{1}{3}n \geq \frac{1}{6}n^3$

Therefore. $f(n) = \frac{1}{3}n^3 - \frac{1}{3}n \geq \frac{1}{6}n^3$. $f(n) = \Omega(n^3)$.

Since $f(n) = O(n^3)$ and $f(n) = \Omega(n^3)$,

$f(n) = \Theta(n^3)$.

The claim follows. ▨.

3.(a).

```
public static int func1(int n) {
   int res = 1;
   for (int i = 1 ; i <= n; i++) {
      for (int j = 2*n; j >= 1; j--) {
         res *= j*j;    // O(1)
      }
      for (int k = 1; k <= n/2; k++) {
         res *= k;    // O(1)
      }
      for (int m = 1; m <= n*n; m++) {
         res *= m;    // O(1)
      }
   }
   return res;
}
```

Inner loops:

For j loop: $j = 2n, 2n-1, 2n-2, \cdots 1$. executed $2n$ times.

For k loop: $k = 1, 2, 3, \ldots \frac{n}{2}$ executed $\frac{n}{2}$ times.

For m loop: $m = 1, 2, 3, \ldots n*n$ executed n^2 times.

All inner loops : we need to add them all, but the low-order terms don't matter.

Thus, the inner loop is executed n^2 times.

The outer loop is executed n times.

The entire code fragment is thus $\Theta(n^3)$.

(b).

```
public static int func2(int n) {
   int sum = 0;
   for (int i = 1; i <= 4n; i *= 2)
         sum++;  // O(1)
   return sum;
}
```

When n=1,	i=1, 2, 4	(i <=4).	$3 = \lfloor \log_2 4 \rfloor + 1$
n=2,	i=1, 2, 4, 8	(i<= 8)	$4 = \lfloor \log_2 8 \rfloor + 1$
n=3,	i=1, 2, 4, 8	(i<= 12)	$4 = \lfloor \log_2 12 \rfloor + 1$
n= 4,	i=1, 2, 4, 8, 16	(i<=16)	$5 = \lfloor \log_2 16 \rfloor + 1$
n= 5,	i=1, 2, 4, 8, 16	(i<= 20)	$5 = \lfloor \log_2 20 \rfloor + 1$
n= 6,	i=1, 2, 4, 8, 16	(i<= 24)	$5 = \lfloor \log_2 24 \rfloor + 1$
n= 7	i=1, 2, 4, 8, 16	(i<= 28)	$5 = \lfloor \log_2 28 \rfloor + 1$
n= 8	i=1, 2, 4, 8, 16, 32	(i<= 32)	$6 = \lfloor \log_2 32 \rfloor + 1$
⋮	⋮	⋮	⋮
			$\lfloor \log_2 4n \rfloor + 1$

We doubling i until it reaches $4n$, so, $2^k = 4n$, where k is the number of times the loop is executed. Take the log of each side, we have $k = \log_2 4n = \log_2 4 + \log_2 n = 2 + \log_2 n$.

Thus, the loop is executed $\lg n$ times, and the code fragment takes time $\Theta(\lg n)$.

(c).

```
public static int func3(int n) {
   int sum = 0;
   for ( int i = 0 ; i < n ; i++)          //executed n times.
       for (int  j = 0 ; j < n*n*n; j++)
            for( int k = 0; k < j ; k++)
                sum++;    // O(1)
   return sum;
}
```

middle loop: $j = 0, 1, 2, 3, \cdots n^3-1$, executed n^3 times.

inner loop: k despends on j.

$j=0$.		0
$j=1$.	$k=0$,	1
$j=2$,	$k=0,1$.	2
$j=3$,	$k=0,1,2$,	3.
⋮		
$j=n^3-1$		n^3-1.

When j is (n^3-1), k can be n^3-1 numbers.

The time the middle loop and inner loop is executed:

$$\sum_{j=0}^{n^3-1} j = 0+1+2+\cdots+(n^3-1)$$

$$= \frac{n^3(n^3-1)}{2} \in \Theta(n^6).$$

Thus, the code fragment is $\Theta(n^7)$.