Problem 1

(a)

3.1-2:

Let $f(n) = (n+a)^b$, and we define $f(n) = \Theta(g(n))$. From the definition of Θ. We know that there exists c_1, c_2 and n_0, which let any $n \geq n_0$, $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$.
By Binomial theorem, we can know that,

$$(n+a)^b = C_b^b n^b + C_b^{b-1} n^{b-1} a + ... + C_b^0 a^b$$

So we can easily find that for $c_1 = 1$, $c_2 = b$, $g(n) = n^b$, the definition is true. So $(n+a)^b = \Theta(n^b)$.

3-2:

A	B	O	o	Ω	ω	Θ
$lg^k n$	n^ϵ	yes	yes	no	no	no
n^k	c^n	yes	yes	no	no	no
$\sqrt{n}$	$n^{sin(n)}$	no	no	no	no	no
2^n	$2^{n/2}$	no	no	yes	yes	no
n^{lgc}	c^{lgn}	yes	no	yes	no	yes
$lg(n!)$	$lg(n^n)$	yes	no	yes	no	yes

3.2-7:

While $i = 0$, we can easily know that,

$$F_0 = 0$$

While $i = 1$, we can find that,

$$\begin{aligned} F_1 &= \frac{\frac{1+\sqrt{5}}{2} - \frac{1-\sqrt{5}}{2}}{\sqrt{5}} \\ &= \frac{\sqrt{5}}{\sqrt{5}} = 1 \end{aligned}$$

While $i = k$, we can find that,

$$F_k = \frac{\phi^k - (\overline{\phi})^k}{\sqrt{5}}$$

While $i = k+1$, we can find that,

$$\begin{aligned} F_{k+1} &= F_k + F_{k-1} \\ &= \frac{\phi^k - (\overline{\phi})^k}{\sqrt{5}} + \frac{\phi^{k-1} - (\overline{\phi})^{k-1}}{\sqrt{5}} \\ &= \frac{(1+\frac{\sqrt{5}-1}{2})\phi^k - (1-\frac{\sqrt{5}+1}{2})(\overline{\phi})^k}{\sqrt{5}} \\ &= \frac{\frac{1+\sqrt{5}}{2}\phi^k - \frac{1-\sqrt{5}}{2}(\overline{\phi})^k}{\sqrt{5}} \\ &= \frac{\phi^{k+1} - (\overline{\phi})^{k+1}}{\sqrt{5}} \end{aligned}$$

So we have prove that

$$F_i = \frac{\phi^i - (\overline{\phi})^i}{\sqrt{5}}$$

(b)

$$O(\prod_{k=1}^{n}(1-\frac{1}{k^2})) = O((1-\frac{1}{n})^n) = O(n^{\frac{1}{n}}) \in O(ln(lnn)) \in O(lnn) = O(\sum_{k=1}^{n}\frac{1}{k}) \in O(n^{1+cosn}) \in O(log(n!)) \in O(3^{lnn}) \in O(n^2+3nlnn+5) = O(n^2+3n+5) = O(n^2+n^{-2}) = O(\sum_{k=1}^{logn}\frac{n^2}{2^k}) \in O(2^n) \in O(n!) \in O((1+n)^n) \in O(n^{n^2-1}) \in O(n^{n^2}+n!)$$

(c)

In this question, we can just take $T(n) = c_1n + c_2nlog_2(n)$ into $T(n) = 2T(\frac{n}{2}) + n$, then we can get,

$$\begin{aligned} c_1n + c_2nlog_2(n) &= 2*(c_1\frac{n}{2} + c_2\frac{n}{2}log_2(\frac{n}{2})) + n \\ c_1n + c_2nlog_2(n) &= c_1n + c_2nlog_2(\frac{n}{2}) + n \\ c_1n + c_2nlog_2(n) &= c_1n + c_2n(log_2n - log_22) + n \\ c_1n + c_2nlog_2(n) &= (c_1 - c_2 + 1)n + c_2nlog_2(n) \end{aligned}$$

From this, we can know that $c_2 = 1$ and c_1 can be any real numbers.

(d)

For the new case, we just do it again.

$$\begin{aligned} c_1 n^\gamma + c_2 n^k &= b^\gamma * (c_1(\frac{n}{b})^\gamma + c_2(\frac{n}{b})^k) + n^k \\ &= c_1 n^\gamma + c_2 b^{\gamma-k} n^k + n^k \\ &= c_1 n^\gamma + (c_2 b^{\gamma-k} + 1) n^k \end{aligned}$$

So, c_1 can be any real numbers and $c_2 = \frac{1}{1-b^{\gamma-k}}$. If we set $\gamma = k$, it will be wrong. That is because the denominator of c_2 will be 0.

Then we start over with $T(n) = c_1 n^\gamma + c_2 n^\gamma log_2 n$.

$$\begin{aligned} c_1 n^\gamma + c_2 n^\gamma log_2 n &= b^\gamma * (c_1(\frac{n}{b})^\gamma + c_2(\frac{n}{b})^\gamma log_2(\frac{n}{b})) + n^\gamma \\ &= c_1 n^\gamma + c_2 n^\gamma (log_2 n - log_2 b) + n^\gamma \\ &= (c_1 + 1) n^\gamma + c_2 n^\gamma (log_2 n - log_2 b) \\ &= (c_1 + 1 - c_2 log_2 b) n^\gamma + c_2 n^\gamma log_2 n \end{aligned}$$

So, c_1 can be any real numbers and $c_2 = \frac{1}{log_2 b}$.

Problem 2

Algorithm 1 Nuts and Bolts

Input: $Arraynuts, Arraybolts, begin, end$
Output: $Arraynuts, Arraybolts$

```
0: function SortNutAndBolt(Arraynuts, Arraybolts, begin, end)
  if begin > end then
    return
  end if
  tempb = Arraybolts[end]
  ini = begin
  for i in begin to end − 1 do
    if Test(Arraynuts[i], tempb) < 0 then
      temp = Arraynuts[i]
      Arraynuts[i] = Arraynuts[ini]
      Arraynuts[ini] = temp
      ini + +
    else if Test(Arraynuts[i], tempb) == 0 then
      temp = Arraynuts[i]
      Arraynuts[i] = Arraynuts[end]
      Arraynuts[end] = temp
      i − −
    end if
  end for
  temp = Arraynuts[ini]
  Arraynuts[ini] = Arraynuts[end]
  Arraynuts[end] = temp
  tempn = Arraynuts[ini]
  ini = begin
  for j in begin to end − 1 do
    if Test(tempn, Arraybolts[j]) < 0 then
      temp = Arraybolts[j]
      Arraybolts[j] = Arraybolts[ini]
      Arraybolts[ini] = temp
      ini + +
    else if Test(tempn, Arraybolts[j]) == 0 then
      temp = Arraybolts[j]
      Arraybolts[j] = Arraybolts[end]
      Arraybolts[end] = temp
      j − −
    end if
  end for
  temp = Arraybolts[ini]
  Arraybolts[ini] = Arraybolts[end]
  Arraybolts[end] = temp
  SortNutAndBolt(Arraynuts, Arraybolts, begin, ini − 1)
  SortNutAndBolt(Arraynuts, Arraybolts, ini + 1, end)
```