Problem 1

```java
public static int function(int n){
    if (n == 1 || n == 2) {
        return 1;
    }
    if (n > 2) {
        return function(n - 1) + 1;
    }
    return -1;
}
```

$$T(1) = 2$$
$$T(2) = 2$$
$$T(3) = 2 + 1 + T(2) + 1 = T(2) + 4 = 6$$
$$T(4) = T(3) + 4 = 10$$
$$\dots\dots$$

If $n = 1$, $T(n) = 2$

If $n \geq 2$, $T(n) = 4n - 6$

Porblem 2

(a)

$f(n) = \sqrt{n} + \sin n$ and $g(n) = \sqrt{n} + n$

For large n, $\sin n$ oscillates between -1 and 1, which means it is bounded and does not grow as n increases.

Therefore, the $\sin n$ term is negligible compared to $\sqrt{n}$ and n.

Formally, we choose $c = 1, n_0 = 1$

$f(n) \leq 1 \cdot g(n)$, when $n \geq 1$

So $f(n) = O(g(n))$

(b)

$f(n) = 2n$ and $g(n) = \log^k n$

We choose $c = 1, n_0 = k$

When $n = n_0 = k$, $f(n) = 2k, g(n) = k, f(n) \geq 1 \cdot g(n)$

So $f(n) = \Omega(g(n))$

We choose $c = 2, n_0 = k$

When $n = n_0 = k$, $f(n) = 2k, g(n) = k, f(n) \leq 2 \cdot g(n)$

So $f(n) = O(g(n))$

So $f(n) = \Theta(g(n))$

(c)

$f(n) = 4n^2 + 3n + 2$ and $g(n) = 2n^2 + 3n + 4$

We choose $c = \frac{1}{2}, n_0 = 1$

$4n^2 + 3n + 2 = f(n) \geq \frac{1}{2}g(n) = n^2 + 1.5n + 2$, when $n \geq 1$

So $f(n) = \Omega(g(n))$

We choose $c = 2, n_0 = 1$

$4n^2 + 3n + 2 = f(n) \leq \frac{1}{2}g(n) = 4n^2 + 6n + 8$, when $n \geq 1$

So $f(n) = O(g(n))$

So $f(n) = \Theta(g(n))$

Problem 3

(a)

$$f(n) + g(n) = \Theta(2f(n) + 2g(n))$$

Let $h(n) = f(n) + g(n)$, question is whether $h(n) = \Theta(2 \cdot h(n))$

Because $f(n) > 0, g(n) > 0$, so $f(n) + g(n) = h(n) > 0$

Let $c = 1$, $h(n) \leq 1 \cdot 2h(n)$, (because $h(n) > 0$)

So $h(n) = O(2 \cdot h(n))$

Let $c = \frac{1}{2}$, $h(n) \leq \frac{1}{2} \cdot 2h(n)$, (because $h(n) > 0$)

So $h(n) = \Omega(2 \cdot h(n))$

So $h(n) = \Theta(2 \cdot h(n))$

(b)

$$f(n) \times g(n) = \Theta(f(2n) \times g(2n))$$

Let $f(n) = g(n) = 2^n$, the question is whether $2^{2n} = \Theta(2^{4n})$

$\lim_{n\to\infty} \frac{2^{4n}}{2^n} = \lim_{n\to\infty} 2^{3n} = \infty$

So $f(n) \times g(n) \neq \Omega(f(2n) \times g(2n))$

So $f(n) \times g(n) \neq \Theta(f(2n) \times g(2n))$

Problem 4

$$T(n) = 3T(\frac{n}{3}) + n$$

$$T(n) = 3T(\frac{n}{3}) + n$$

$$T(n) = 3(3T(\frac{n}{3^2}) + \frac{n}{3}) + n$$

$$T(n) = 3(3(3T(\frac{n}{3^3}) + \frac{n}{3^2}) + \frac{n}{3}) + n$$

$$\dots\dots$$

$$T(n) = 3^{\log_3 n} \cdot 1 + 3^{\log_3 n-1} \cdot 3 \cdots + n + n = n\log_3 n + n$$

So $T(n) = nlog_3n + n$

Problem 5

From the question, we can set $T(n) = T(\frac{a}{a+b}n) + T(\frac{b}{a+b}n) + n$. And $T(1) = 1$.

The answer is $T(n) = \Theta(n\log n)$

We can use Akra–Bazzi method to prove it:

$$T(x) = g(x) + \sum_{i=1}^{k} a_i T(b_i x + h_i(x)) \qquad \text{for } x \geq x_0 . \sum_{i=1}^{k} a_i b_i^p = 1$$

$$T(x) \in \Theta\left(x^p\left(1 + \int_1^x \frac{g(u)}{u^{p+1}} du\right)\right)$$

In this case, $p = 1$, and $g(u) = u$

We can calculate

$$\Theta(n^p(1 + \int_1^n \frac{g(u)}{u^{p+1}} du))$$
$$= \Theta(n^1(1 + \int_1^n \frac{u}{u^{1+1}} du))$$

$$= \Theta(n(1 + \int_1^n \frac{u}{u^2} du))$$

$$= \Theta(n(1 + \int_1^n \frac{1}{u} du))$$

$$= \Theta(n(1 + \log n - \log 1))$$

$$= \Theta(n(1 + \log n))$$

$$= \Theta(n + n \log n)$$

$$= \Theta(n \log n)$$

Problem 6

---

1)

```
FUNCTION mergeItems(items1, items2):
    // Step 1: Initialize an empty dictionary to store weights summed
    DECLARE weight_map AS DICTIONARY

    // Step 2: Iterate over items1 and populate weight_map
    FOR EACH (value, weight) IN items1:
        IF value IN weight_map:
            weight_map[value] ← weight_map[value] + weight
        ELSE:
            weight_map[value] ← weight

    // Step 3: Iterate over items2 and populate weight_map
    FOR EACH (value, weight) IN items2:
        IF value IN weight_map:
            weight_map[value] ← weight_map[value] + weight
        ELSE:
            weight_map[value] ← weight

    // Step 4: Convert the weight_map to a list of tuples
    DECLARE ret AS LIST
    FOR EACH key, value IN weight_map:
        APPEND (key, value) TO ret

    // Step 5: Sort the list of tuples by value in ascending order
    SORT ret BY key IN ASCENDING ORDER

    // Step 6: Return the sorted list
    RETURN ret
```

2)

Building `weight_map`: $O(n + m)$

Sorting `weight_map`: $O(k \log k)$, where (k) is the number of unique values in `items1` and `items2`.

The combined time complexity is $O(n + m + k \log k)$