1.(a) O(1)
(b) Θ(n + m)
(c) O(n^2 * logn)
(d) O(n)
(e) O(n + N)
(f) Ω(d)
(g) O(n)
(h) O(n^2)
(i) O((n + m) * log(logn))

2.

Proof : Since u.d < v.d, it means that vertex u is discovered before vertex v. Also, there is at least one path from u to v. Therefore, things must be that in DFS, vertex u is discovered and then from the edges connected to u, vertex v is discovered. Therefore, v is a descendant of u in corresponding DFS forest.

3. (a) False
(b) (n^2 - m)
(c) the number of different value of characters in this digit
(d) False
(e) False
False
(f) three, which is 10, 24 and 03.
(g) True.
(h) Zk is an LCS of X and Yn-1.

4.

The error of the algorithms is at the second line. Instead, we should choose the min-weight edge which connect from the vertex in the MST to the vertex not in the MST.
For example, in undirected graph

The vertices are A B C D E and the edges and weight are (A,B) = 1, (A,D) = 2, (B,D) = 4, (B, C) = 4, (B, E) = 3 , (C, E) = 5,(C,D) = 2, (D, E) = 4.

For the algorithms, first pick A and add (A,B) = 1; then pick B add (B,E) = 3; then pick E and add (D,E) = 4; finally pick D and add (D,C) = 2. The total weight is 10.

However, there is another tree, which is (A,B) = 1, (A,D) = 2, (D,C) = 2, (B,E) = 3. The total weight is 8.

Therefore, the algorithms is wrong.

5.

```
Solution(A, S)
{
        boolean res[][] = new boolean[A.length][S+1];
        boolean memo[][] = new boolean[A.length][S+1];
        for i = 0 to A.length
                for j = 0 to S
                        res[i][j] = false;
                        memo[i][j] = false;
        return findSet(A, S, A.length, res, memo);
}
```

```
findSet(A, S, end, res, memo)
{
	if(end == 0)
	{
		res[end][S] = (A[end] == S);
		memo[end][S] = true;
	}

	if memo[end][S] = true;
		return res[end][S];

	res[end][S] = findSet(A, S, end - 1, res, memo) || findSet(A, S - A[end], end - 1, res,
memo);
	memo[end][S]  = true;
	return res[end][S];
}
```

6.

```
RABIN - KARP(T[0…n -1], P[0…m - 1], d)
{
	Let q to be an prime number.
	int n = T.length
	int m = P.length
	int h = d^(m -1) mod q
	int t = 0;
	int p = 0;
	for i = 0 to m
		t = (d * p + T[i]) mod q;
		p = (d * t + P[i] mod q);
	for i = 0 to n - m
		if(p == t)
			int k = 0;
			while(k < m)
				if(P[k] != T[i + k])
					break;
			if(k == m)
				print i;
		if i < n - m
			t = ((d * (t - T[i]) * h) + T[i + m]) mod q;
}
```

7.

```
// Assume that the vertices are labeled as the number of sequence of 0, 1, 2, 3 … n -1
// G.AdjList[0] means the head node of list of vertices connected to vertex 0.
// G.AdjList[0].next means the next node of the list.
// G.AdjList[0].val means the label value of the node
vertexDegrees(G.AdjList, m,n)
{
	int outDegree[] = new int[n];
	int inDegree[] = new int[n];
	for i = 0 to n - 1
```

```
            outDegree[i] = 0;
            inDegree[i] = 0;
        for i = 0 to n - 1
            node p = G.AdjList[i];
            while(p != NIL)
                outDegree[i] = outDegree[i] + 1;
                inDegree[p.val] = inDegree[p.val] + 1;
                p = p.next;
        for i = 0 to n - 1
            System.out.print(outDegree[i]);
            System.out.print("  ");
            System.out.print(inDegree[i]);
            System.out.println(" ");

}
```

running time : initialization: Θ(n) since we iterate all the vertices

calculation : Θ(m) since we iterate all the edges of the graph.

total running time : Θ(n + m)

8.

I will use counterexample to prove that the greedy method is wrong.

counterexample : Assume that there are three items total which are labeled as 1, 2, 3. and the most W pounds is 100. The value of 1 is 15 and its weight is 51; the value of 2 and 3 is 10 and its weight is 50;

For greedy strategy, thief will choose the maximum ratio. 15/51 = 0.29, 10/50 = 0.2. So the thief will choose the item 1 and he can only can choose item one since 51 + 50 > 100. Therefore he take 15 value away.

However, the optimal solution is to take item 2 and item 3. The weight is 100 which can be carried and the value of the two items is 10 + 10 = 20 which is bigger than 15.

Therefore, the greedy strategy in the question is not the optimal solution.