HW2 TA Reference Solutions

(Reference answers are based on textbook and online resources.)

1. (10 points) Problem 2 on page 143.

No, it's not a good message digest function. If we interchange the chunks, we would get the same hash. Different messages may generate same digest, as long as they are generated with the same 128-bit XOR.

2, (10 points) Problem 6 on page 144.

If not using this padding, it would be easy to find two messages with the same hash. Assuming M' is a message that is not a multiple of 512 bits, and M' is M padded as with MD4 (so M is a multiple of 512 bits). If no padding is used for M, MD4(M) would be the same as MD4(M').

3. (10 points) Problem 9 on page 144.

MD5: min: 1 bit; max: 512 bits
SHA: min: 1 bit; max: 512 bits
(The padding needs 512 bits for the latter three.)

4. (10 points) Problem 12 on page 145.

No, in either way the number of messages is 2^128

5. (20 points; 2.5 points per item) Problem 14 on page 145.

(The question means giving the minimal sufficient conditions. You mainly get points as long as the condition in your answer is sufficient. But they may not complete)
~ x : X must be random
x⊕ y : at least one of x or y must be random
x ∨ y : at least one of x or y must be random, and let the other one be zero
x ∧ y : at least one of x or y must be random, and let the other one be all ones
(x∧ y)∨ (~ x∧ z) : let one of (x AND y, ~x AND z) be zero and the other be random -- so if x is all ones and y is random, then the whole expression is random
(x ∧ y) ∨ (x ∧ z) ∨ (y ∧ z): we can set two of the clauses to be zero and let the other be random. So, set z to be zero, y to be all ones, and let x be random
x⊕ y⊕ z: at least one of x, y, z must be random.
y⊕ (x ∨−z) : at least one of x, y, z must be random.

6. (10 points) Problem 18 on page 145.

The decryption process:
b1 = MD(KAB |IV); p1 = c1 ⊕ b1
b2 = MD(KAB |c1); p2 = c2 ⊕ b2
. . .
bi-1=MD(KAB|ci-2)
bi = MD(KAB |ci−1); pi = ci ⊕ bi

KAB, ci and IV are given, then calculate bi.
Then we have pi.

7. (10 points) Problem 19 on page 146.

After the modification:

b1 = MD(KAB |IV); c1 = p1 ⊕ b1
b2 = MD(KAB |p1); c2 = p2 ⊕ b2

. . .
bi = MD(KAB |pi−1); ci = pi ⊕ bi

We can decrypt using:

b1 = MD(KAB |IV); p1 = c1 ⊕ b1
b2 = MD(KAB |p1); p2 = c2 ⊕ b2

. . .
bi = MD(KAB |pi−1); pi = ci ⊕ bi

If the plaintext consisted of all zeros, it's easy to calculate ci and then get the key. It's not secure.