SAT: Structural Testing and Code Coverage

Consider Java's implementation of the LinkedList's `computeIfPresent()` method:

```
1  public V computeIfPresent(K key,
2                            BiFunction<? super K, ? super V, ? extends V> rf) {
3    if (rf == null) {
4      throw new NullPointerException();
5    }
6    Node<K,V> e;
7    V oldValue;
8    int hash = hash(key);
9    e = getNode(hash, key);
10   oldValue = e.value;
11   if (e != null && oldValue != null) {
12     V v = rf.apply(key, oldValue);
13     if (v != null) {
14       e.value = v;
15       afterNodeAccess(e);
16       return v;
17     } else {
18       removeNode(hash, key, null, false, true);
19     }
20   }
21   return null;
22 }
```

1. What is the minimum number of tests needed for 100% (and why):

a) line coverage? 3 (some conditions are mutually exclusive)

b) branch coverage? 4 (most nested needs 2 + 1 x 2 for each outer)

c) branch+condition coverage? 4/5 (one branch has 2 conditions)

d) path coverage? 16 (2^4)

e) MC/DC? 5 (4 + 1)

Consider the expression (A & B) || C with the following truth table:

Test case	A	B	C	(A & B) | C
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	T
6	F	T	F	F
7	F	F	T	T
8	F	F	F	F

2. What test suite(s) achieve(s) 100% MC/DC?

```
A - {2,6}
B - {2,4}
C - {3,4}, {5,6}, {7,8}
Either {2,3,4,6} or {2,4,5,6}
```

3. Draw the truth table for the expression A & (A||B). What test suite(s) achieve 100% MC/DC? What can you say about this piece of code?

```
A - {1,3}, {2,4}
B - none
Either {1,3}, {2,4}
Expression can be simplified to A
```