A. (a) Any element can only increase in depth within its tree when it is in the smaller of two sets being UNIONed, and hence at most $\lfloor \lg n \rfloor$ times, each time gaining in depth by 1. Thus the find path of any element will be of length $O(\lg n)$. Hence each UNION and each FIND-SET operation takes $O(\lg n)$ time, while each MAKE-SET operation executes in $O(1)$ time.

Therefore, executing m disjoint-set operations, n of which are MAKE-SET operations, takes $O(m \lg n)$ time.

(b) Assume that $n = 2^k$ and $m \geq 3n$. We first perform n MAKE-SET operations to produce 2^k trees of height 0. We then pairwise combine these with 2^{k-1} UNION operations to create 2^{k-1} trees of height 1. We repeatedly perform this pairwise combining to produce a single tree of depth $k = \lg n$ after a total of $n-1$ UNION operations.

Finally, we perform $m-2n+1$ FIND-SET operations on the deepest element, each of which taking $\Omega(\lg n)$ time. These FIND-SET operations alone thus take a total of $\Omega(m \lg n)$ time.

B. The simple change to the DFS algorithm is to add the following test between lines 6 and 7 of DFS-VISIT (as presented in the lecture slides):

$6\frac{1}{2}$ **else if** $v \neq \pi[u]$ **then** halt and report a cycle.

(a) If the graph contains a cycle, then DFS-VISIT will encounter an edge satisfying the above test when it examines the last vertex on this cycle.

Conversely, if this test is satisfied, it indicates that a cycle has been encountered, namely the DFS tree path from $\pi[u]$ to v, closed by the edge from v to u which satisfied the test.

(b) The running time is $O(V)$ and not $O(V + E)$, since for as long as this test is not satisfied, each edge must introduce a new vertex; hence if the algorithm ever sees $|V|$ distinct edges, then one of these must satisfy the test and cause the algorithm to halt.