In our textbook, Theorem 7.44, Problem 7.26, and Problem 7.27 cover this material.

Definitions: For an undirected graph G, a *vertex cover* is a set of nodes such that every edge is adjacent to at least one node in the set. Let *VERTEX-COVER* be the set of all strings $\langle G, \ell \rangle$, where $\ell \geq 1$ is an integer and G is an undirected graph that has an ℓ-node vertex cover.

Here is a polynomial-time mapping reduction F of $3SAT$ to *VERTEX-COVER*. Given the encoding $\langle \phi \rangle$ of a Boolean formula ϕ in 3CNF, F does these steps:

1. Let m be the number of distinct variables in ϕ, and let k be the number of clauses. Allocate space for a graph G of $2m + 3k$ nodes and $m + 6k$ edges.
2. For each distinct variable x in ϕ, add nodes labeled x and $\overline{x}$, and join these two nodes with an edge to form a "pair". (So this step adds $2m$ nodes and m edges total.)
3. For each clause $(x \vee y \vee z)$ in ϕ, add three nodes with those labels, and join these three nodes with three edges to form a "triangle". (Here, each of x, y, and z can be any variable or its negation. This step adds $3k$ nodes and $3k$ edges.)
4. For each node among the triangles, connect that node to the unique node among the pairs that has the same label. (This step adds $3k$ edges.)
5. Let $\ell = m + 2k$, and output $\langle G, \ell \rangle$.

A.A. Pick a satisfiable ϕ, show the G and ℓ that F produces from that ϕ, and identify an ℓ-node vertex cover in G. (So it's plausible that $\langle \phi \rangle \in 3SAT \Rightarrow F(\langle \phi \rangle) \in \mathit{VERTEX\text{-}COVER}$.)

A.B. Pick an unsatisfiable ϕ, show the G and ℓ produced, and explain why no ℓ-node vertex cover exists. (So it's plausible that $\langle \phi \rangle \in 3SAT \Leftarrow F(\langle \phi \rangle) \in \mathit{VERTEX\text{-}COVER}$.)

A.C. Show that the length of the string $\langle G, \ell \rangle$ is polynomial in the length n of the string $\langle \phi \rangle$. (So, because F "doesn't have to do much thinking beyond what's needed to write $\langle G, \ell \rangle$", it's plausible that F is polynomial-time.)

Definitions: Let ϕ be a 3CNF formula. An $\neq$-*assignment* is an assignment of truth values to the variables in ϕ, such that each clause has one true term (and two false terms) or two true terms (and one false term). Let $\neq SAT$ be the set of strings $\langle \phi \rangle$, where ϕ is a 3CNF formula that has an $\neq$-assignment. Notice that $\neq SAT \subseteq 3SAT$.

Here is a polynomial-time mapping reduction F from $3SAT$ to $\neq SAT$. Given the encoding $\langle \phi \rangle$ of a Boolean formula ϕ in 3CNF, F does these steps:

1. Let k be the number of clauses in ϕ. Allocate space for a new 3CNF formula ψ that has $2k$ clauses, all of the variables of ϕ, and $k+1$ new variables, which are called $w_1, w_2, \ldots, w_k, b$.

2. For $i = 1, \ldots, k$, let $(x \vee y \vee z)$ be the ith clause in ϕ. (Here, each of x, y, and z can be any variable or its negation.) Add two clauses $(x \vee y \vee w_i) \wedge (\overline{w_i} \vee z \vee b)$ to ψ.

3. Output $\langle \psi \rangle$, where ψ is the 3CNF formula of $2k$ clauses that was just produced.

B.A. Pick a satisfiable ϕ, show the ψ that F produces from that ϕ, and identify an $\neq$-assignment for ψ.

B.B. Pick an unsatisfiable ϕ, show the ψ produced, and explain why ψ has no $\neq$-assignment.

B.C. Prove that the logical negation of any $\neq$-assignment is also an $\neq$-assignment. (This problem is relatively easy, but it's useful for the next part of the problem.)

B.D. Prove that $\langle \phi \rangle \in 3SAT \Leftrightarrow F(\langle \phi \rangle) \in {\neq}SAT$.

The formula ψ is about twice as large as the formula ϕ, and F doesn't need to do much thinking to construct ψ, so F should be polynomial-time. Let's not analyze the time complexity of F in more detail than that.

Definitions: In an undirected graph G, a *cut* is a partition of the nodes into two disjoint subsets S and T. The *size* of a cut is the number of edges that have one endpoint in S and the other in T. (Imagine drawing a line L through G and rearranging the nodes so that the S-nodes are on one side of L and the T-nodes are on the other side. Then the size of the cut is the number of edges that L "cuts".) Let $MAX\text{-}CUT$ be the set of strings $\langle G, \ell \rangle$ such that $\ell \geq 1$ is an integer and G has a cut of size ℓ or greater.

Here is a polynomial-time mapping reduction F from ${\neq}SAT$ to $MAX\text{-}CUT$. Given the encoding $\langle \phi \rangle$ of a Boolean formula ϕ in 3CNF, F does these steps:

1. Let m be the number of distinct variables in ϕ, and let k be the number of clauses. Allocate space for a graph G of $6mk$ nodes and $9mk^2 + 3k$ edges.

2. For each distinct variable x in ϕ: Add $3k$ nodes labeled x and $3k$ nodes labeled $\overline{x}$, and add an edge connecting each x-node to each $\overline{x}$-node. (Total: $6mk$ nodes and $9mk^2$ edges.)

3. For each clause $(x \vee y \vee z)$ in ϕ, add three edges connecting nodes labeled x, y, z into a triangle. Do not use any node in more than one clause triangle. (Total: $3k$ edges.)

4. Let $\ell = \ldots$, and output $\langle G, \ell \rangle$.

C. In the last step, what should ℓ be? Also, prove that $\langle \phi \rangle \in {\neq}SAT \Leftrightarrow F(\langle \phi \rangle) \in MAX\text{-}CUT$. (I recommend, but do not require, that you first do a couple of examples to build intuition, as in our previous problems.)

Looking for more practice with polynomial-time mapping reductions? Study the reduction of $3SAT$ to $SUBSET\text{-}SUM$ in our textbook. These reductions get easier, as you see more examples!