Problem Set 9 — Due Wednesday, June 5

Turn this in by **2:45 pm in the box in Kemper**, or by **3:10 pm in discussion section**.

Problem 1. Recall our C program that printed itself:

```
char*f="char*f=%c%s%c;main(){printf(f,34,f,34,10);}%c";main(){printf(f,34,f,34,10);}
```

Modify the program so that it begins and ends with a comment:

```
/* ECS 120 */
...
/* is cool */
```

Your program will now be three lines long. Of course it must still print itself. Compile and run it to make sure it works.

Problem 2. Consider the following two languages:

$CLIQUE = \{\langle G, k\rangle : \ G = (V, E)$ is a graph that contains a clique of size $k\}$, and
$INDSET = \{\langle G, l\rangle : \ G = (V, E)$ is a graph that contains an independent set of size $l\}$.

Above, a *clique* K of size k is in a graph $G = (V, E)$ is a set $K \subseteq V$ of $k = |K|$ mutually connected vertices; while an *indpendent set* L of size l is a set $L \subseteq V$ of $l = |L|$ vertices none of which is connected to any other in L.

Show that $CLIQUE \leq_{\text{P}} INDSET$. This means that you must give a many-one reduction f from *CLIQUE* to *INDSET* that is, additionally, computable in polynomial time.

Problem 3. Suppose you are given a unit-time procedure D that decides *CLIQUE*; that is, $D(\langle G, k\rangle)$ returns 1 if G has a clique of size k; otherwise, it returns 0. Show how to use D to make a polynomial-time algorithm A that finds a largest clique in a graph. That is, algorithm A, on input $\langle G\rangle$, makes subroutine calls to D, which count as taking one time step, and it returns a set $K \subseteq V$ that is a largest clique in $G = (V, E)$.

Problem 4. Let $MYSAT = \{\langle \phi\rangle : \phi$ is a Boolean formula with at least 100 satisfying assignments and at least 100 non-satisfying assignment$\}$. Prove that *MYSAT* is NP-complete.

Problem 5. *Problem 7.28 from your book.* You are given a box and a collection of cards. Because of the pegs in the box and the notches in the cards, each card will fit in the box in either of two ways. Each cards contains two columns of holes, some of which may not be punched out. The puzzle is solved by placing all the cards in the box so as to completely cover the bottom of the box (ie, every hole position is blocked by at least one card that has no hole there). Let $PUZZLE = \{\langle c_1, \ldots, c_k\rangle |$ each c_i represents a card and this collection of cards *has* as solution$\}$. Show that *PUZZLE* is NP-complete.