1. [KT-Chapter8] You have a set of friends $F$ whom you’re considering to invite, and you’re aware of a set of $k$ project groups, $S_1, \ldots, S_k$, among these friends (these sets need not be disjoint). The problem is to decide if there is a set of $n$ of your friends whom you could invite so that not all members of any one group are invited. Prove that this problem is NP-complete.

2. [KT-Chapter8] Given an undirected graph $G = (V, E)$, a feedback set is a set $X \subseteq V$ with the property that $G - X$ has no cycles. The undirected feedback set problem asks: given $G$ and $k$, does $G$ contain a feedback set of size at most $k$? Prove that the undirected feedback set problem is NP-complete.

3. [KT-Chapter8] We have seen the Interval Scheduling problem in class; here we consider a computationally much harder version of it that we will call Multiple Interval Scheduling. As before, you have a processor that is available to run jobs over some period of time. (E.g. 9 AM to 5 PM.)

People submit jobs to run on the processor; the processor can only work on one job at any single point in time. Jobs in this model, however, are more complicated than we’ve seen in the past: each job requires a set of intervals of time during which it needs to use the processor. Thus, for example, a single job could require the processor from 10 AM to 11 AM, and again from 2 PM to 3 PM. If you accept this job, it ties up your processor during those two hours, but you could still accept jobs that need any other time periods (including the hours from 11 to 2). Now, you’re given a set of $n$ jobs, each specified by a set of time intervals, and you want to answer the following question: For a given number $k$, is it possible to accept at least $k$ of the jobs so that no two of the accepted jobs have any overlap in time? Show that Multiple Interval Scheduling is NP-complete.

4. [KT-Chapter8] The following is a version of the Independent Set problem. You are given a graph $G = (V, E)$ and an integer $k$. For this problem, we will call a set $I \subseteq V$ strongly independent if for any two nodes $v, u \in I$, the edge $(v, u)$ does not belong to $E$, and there is also no path of 2 edges from $u$ to $v$, i.e., there is no node $w$ such that both $(u, w) \in E$ and $(w, v) \in E$. The Strongly Independent Set problem is to decide whether $G$ has a strongly independent set of size $k$. Prove that the Strongly Independent Set problem is NP-complete.

5. [KT-Chapter8] Consider the following problem. You are given positive integers $x_1, \ldots, x_n$, and numbers $k$ and $B$. You want to know whether it is possible to partition the numbers $\{x_i\}$ into $k$ sets $S_1, \ldots, S_k$ so that the squared sums of the sets add up to
at most $B$:

$$\sum_{i=1}^{k} \left( \sum_{x_j \in S_i} x_j \right)^2 \leq B.$$  

Show that this problem is NP-complete.

6. **[KT-Chapter8]** Suppose you’re consulting for a group that manages a high-performance real-time system in which asynchronous process make use of shared resources. Thus, the system has a set of $n$ processes and a set of $m$ resources. At any given point in time, each process specifies a set of resources that it requests to use. Each resource might be requested by many processes at once; but it can only be used by a single process at a time. Your job is to allocate resources to processes that request them. If a process is allocated all the resources it requests, then it is active; otherwise it is blocked. You want to perform the allocation so that as many processes as possible are active. Thus, we phrase the Resource Reservation problem as follows: given a set of process and resources, the set of requested resources for each process, and a number $k$, is it possible to allocate resources to processes so that at least $k$ processes will be active? Show that Resource Reservation is NP-complete.

8. In an undirected graph $G = (V, E)$, we say a subset $D \subseteq V$ is a dominating set if every $v \in V$ is either in $D$ or adjacent to at least one member of $D$. In the DOMINATING SET problem, the input is a graph and a number $b$, and the aim is to find a dominating set in the graph of size at most $b$, if one exists. Prove that this problem is NP-complete.