CS260P HW8

#1
1) Yes. The reductions Interval Scheduling ≤ p Independent Set and Independent Set ≤ p Vertex Cover, so the result follows by transitivity.
2) This is equivalent to whether P=NP. If P=NP, then Independent Set can be solved in polynomial time, and so Independent Set ≤ p Interval Scheduling. Conversely, if Independent Set ≤ p Interval Scheduling can be solved in polynomial time, so could Independent Set.

#2
The problem is in NP because we can exhibit a set of k customers, and in polynomial time can be checked that no two bought any product in common. Given a graph G and a number k, we construct a customer for each node of G, and product for each edge of G. Then we build an array that says customer v bought product e if edge e is incident to node v. We claim that this holds if and only if G has an independent set of size k. If there is a diverse subset of size k, then the corresponding set of nodes has the property that no two are incident to the same edge—so it is and independent set of size k.

#3
The problem is in NP since, given a set of k counselors, we can check that they cover all the sports. Suppose we had such an algorithm A: here is how we would solve an instance of Vertex Cover. Given a graph G=(V,E) and an integer k, we would define a sport Se for each edge e, and a counselor Cv for each vertex v. Cv is qualified in sport Se if e has an endpoint equal to v. Now, if there are k counselors that, together, are qualified in all sports, the corresponding vertices in G have the property that each edge has an end in at least one of them; so they define a vertex cover of size k. Thus, G has a vertex cover of size at most k if and only if the instance of Efficient Recruiting that we create can be solved with at most k counselors.

#4
We first show the problem is in NP. To see this, notice that if we are given a set of k processes, we can check in polynomial time that no resource is requested by more than one of them. To prove that the Resource Reservation in NP-complete we use the independent set problem, which is known to be NP-complete. The case k=2 can be solved by brute force: we just try all O (n2) pairs of processes, and we see whether any pair has disjointed resource requirements. This is a polynomial-time solution. We define a node for each person and each piece of equipment, and each process is an edge joining the person and the piece of equipment it needs. We observe that our reduction in actually created an instance of Resource Reservation that had this special form.

#5
Hitting Set is in NP: Given an instance of the problem, and a proposed set H, we can check in polynomial time whether H has size at most k, and whether some member of each set Si belongs to H. Hitting Set looks like a covering problem, since we are trying to choose at most k objects subject to some constraints. This suggests that we define the set A in the Hitting Set instance to be the V of nodes in the Vertex Cover instance. Now we claim that there is a hitting set of size at most k for this instance, if and only if the original graph had a vertex cover of size at most k. For if we consider a hitting set H of size at most

k as a subset of the nodes of G, we see that every set is “hit”, and hence edge has at least one end in H: H is a vertex cover of G.

#6

We choose Vertex Cover as the problem X, and show Vertex Cover ≤ p Monotone Satisfiability with Satisfiability with Few True Variables. Suppose we are given a graph G=(V,E) and a number k; we want to decide whether there is a vertex cover in G of size at most k.

#7

4-Dimensional Matching is in NP, since we can check in O(n) time, using an n*4 array initialized to all 0, that a given set of n 4-tuples is disjoint. We now show that 3-Dimensional Matching ≤ p 4-Dimensional Matching. So, consider an instance of 3-Dimensional Matching, with sets X, Y, and Z of size n each, and a collection C of ordered triples. We define an instance of 4-Dimensional Matching as follows. Thus, by determining whether there is a prefer 4-Dimensional Matching in the instance we have constructed, we can solve the initial instance of 3-Dimensional Matching.