Due 23:59 Nov 2 (Sunday). There are 100 points in this assignment.
Submit your answers **(must be typed)** in pdf file to CourSys
`https://coursys.sfu.ca/2025fa-cmpt-705-x1/`.
Submissions received after 23:59 will get penalty of reducing points: 20 and 50 points deductions for submissions received at $[00:00, 00:10]$ and $(00:10, 00:30]$ of Nov 3, respectively; no points will be given to submissions after 00 : 30 of Nov 3.

1. (Chapter 8 Problem 1 of text book) 10 points

   For each of the two questions below, decide whether the answer is (i) "Yes," (ii) "No," or (iii) "Unknown, because it would resolve the question of whether $P = NP$." Give a brief explanation of your answer.

   (a) The decision version of the Interval Scheduling Problem from Chapter 4 is defined as follows: Given a collection of intervals on a time-line, and a bound k, does the collection contain a subset of nonoverlapping intervals of size at least k? Question: Is it the case that Interval-Scheduling$\leq_P$Vertex Cover?

   Yes, since Interval-Scheduling is in P, and $P \subseteq NP$ and Vertex Cover is NP-Complete, then Interval-Scheduling$\leq_P$Vertex Cover.

   (b) Unknown, because it would resolve the question of whether $P = NP$.

   No need to say that any two problem in P can be reduced in polynomial time to each other. For instance to show that $X \leq_P Y$, just use no calls to Y and solve X in polynomial time itself.

   If $P = NP$, then there is a polynomial time algorithm to solve Independent-Set. Thus, if $P = NP$, then Independent-Set$\leq_P$Interval-Scheduling.

   Moreover, since Independent-Set is NP-Complete if Independent-Set$\leq_P$Interval-Scheduling, then $P = NP$.

   Hence, iff Independent-Set$\leq_P$Interval-Scheduling, then $P = NP$

2. (Chapter 8 Problem 2 of the text book) 15 points

   (Note: it should be at "**most**" k rows instead of at "**least**" k rows. That is because if at "least" k rows" is requested, we can choose all rows and it can be check in polynomial time if at least one row is not equal to 0.)

   Given a certificate with k rows, it takes $O(k \times n)$ to check if for each column, at least one row is not equal to 0. Thus the problem is in NP.

   I show that Vertex Cover $\leq_P$ this problem.

   Given a vertex cover problem with graph $G(V, E)$, let A be an array with size $|E| \times |V|$ (m columns and n rows). Let for each column i, the two rows $k = j = 1$ iff edge $e_i, 1 \leq i \leq m$, connect two nodes u_k and u_j $1 \leq j, k \leq n$. Otherwise, all rows are zero.

   Assume we find a set of at most k rows that have at least one row larger than 0 for each column. Thus, for each column, there is at least one row that has value equal

to 1. It means for each edge, there is one node that is selected. There are at most k selected rows.

Thus we have found a vertex cover with at most k nodes. In this case, for each edge, there is one nodes that is connected to that edge in the set of k nodes.

If there is not at most k rows that has at least one row equal to 1 for each column, then there is no set of k nodes that covers all edges.

All steps can be done in $O(n \times m)$

Hence the vertex-cover problem is polynomial reducible to this problem and this problem is NP-Complete

3. (Chapter 8 Problem 5 of the text book) 15 points

   Given a certificate H with size k, it takes $O(n \times m)$ to check if $H \cap B_i \neq \emptyset$. That is to check for each element of $B_1, .., B_m$, if they exist in H. (existence can be checked using a boolean array). Thus Hitting Set problem is in NP.

   I show that Vertex Cover $\leq_P$ Hitting Set problem:

   Given a graph G(V,E), define $A = V$ and for each edge e_i, $1 \leq i \leq m$ that connects u_i and u_j, $1 \leq j, k \leq n$, set $B_i = \{u_j, u_k\}$.

   Assume we find a hitting set H with size k that $H \cap B_i \neq \emptyset$. Each B_i has at least one element in H. That means, for each edge, there is at least one adjacent node in set H. H has k elements. H is a vertex cover of size k for graph G.

   All steps takes $O(n + m)$

   Thus, Hitting Set problem is NP-Complete

4. (Chapter 8 Problem 13 of the text book) 15 points

   Given a certificate A, it takes linear time to check $\sum_{(S_i, x_i) \in A} x_i \geq B$ and it takes $O(m \times n)$ to check if $S_i \cap S_j = \emptyset$, that is to define boolean array B_i, with size n for each $S_i, 1 \leq i \leq m$ and check for each B_i at most one element $b_j, 1 \leq j \leq n$ is true. This combinatorial auction problem is in NP.

   I show that Independent Set problem $\leq_p$ combinatorial auction problem:

   Given an independent set problem $G(V, E)$, define $I = V$. Define S_i as the the set of edges connected to node u_i, that is:

$$S_i = \{\, e_j \mid \text{edge } e_j \text{ is connected to node } u_i \,\}$$

   If a node has degree of zero, S_i is an empty set. set $x_1 = ... = x_m = 1$ and $B = k$.

   If combinatorial auction problem is solved, then there is at least k disjoint S_i. If S_i is disjoint from S_j, they have no shared edges, thus they are not connected. If s_i is selected, put u_i in set P. The setP is set of k independent nodes.

   All steps takes $O(n + m)$

Hence the independent set problem is polynomial reducible to the combinatorial auction problem.

5. (Chapter 8 Problem 20 of the text book) 15 points

   Given a certificate $C_1, ..., C_k$ for n points, it takes $O(n^2)$ to calculate distance of each pair of points and find the largest distance within each cluster and eventually check if $d(C_i) \leq B$. Thus, low-diameter clustering problem is in NP.

   I will show that Graph K-Color $<_P$ low-diameter clustering problem

   Given Graph k-color problem $G(V, E)$, let $p_i = u_i$, $u_i \in V$ for $i = 1$ to n. That is define one point for each node.

   define $d(p_i, p_j) = 1$ if there is no edge betweenu_i and u_j and $d(p_i, p_j) = |V| + 1 = n + 1$ if there is an edge between u_i and u_j.

   set $B = |V|$ and partition $p_1, .., p_n$ into k clusters $C_1, .., C_k$. Each cluster is a color. If there is an edge between nodes of each cluster C_i, then $d(C) > |v|$, other wise $d(C) \leq n \times 1 = |V|$. This guaranty that there is no edge between nodes in the same color. Since there are k clusters, it is guaranteed that there are at most k colors.

   If there is not such a clusters, the graph cannot be not k-color satisfiable.

   All steps takes $O(n + m)$

6. (Chapter 8 Problem 21 of the text book) 15 points

   Given a certificate with a set of k elements S_k, it takes $O(n \times k)$ to if one element from each $A_1, .., A_k$ exist in s_k. (sets are disjoint, so there could not be more than one elements). Thus the FCC problem is in NP.

   I will show that 3-SAT Problem $<_p$ FCC problem.

   For a 3-CFN instance $\phi = C_1...C_k$, let v_{i1}, v_{i2}, v_{i3} be the three literals in clause C_i. Two literals v_{ij} and $v_{i'j'}$ are in conflict if one of them is variable x and the other is negation of x ($\bar{x}$).

   I construct set P as the set of literals in conflict and A_i set of literals v_{i1}, v_{i2}, v_{i3} in clause C_i. $A_1, .., A_k$ are disjoint sets.

   If two conflicting literals are in one clause, then this clause is always satisfied. Thus remove that clause from the problem. Thus, if $(v_{ij}, v_{i'j'}) \in P$, $v_{ij} \in A_i$ and $y \in v_{i'j'}$ with some $1 \leq i \neq i' \leq k$

   If the FCC problem is solved, we can select one element vij from each A_i so that its negation $v_{i'j'}$ is not selected. Thus, set the selected element v_{ij} from each A_i equal to 1. There is no conflict and all clause are satisfied. Thus the formula is satisfied.

   If this selection is not feasible, then we cannot select even one element from each clause that is not in conflict with another element from another clause. (If we cannot select one element, we cannot select more than one element). It means, there is at least one clause that is not satisfied. Thus the formula is not satisfiable.

All steps takes $O(n + k)$

Thus, 3-SAT problem is polynomial reducible to FSS problem.

7. (Chapter 9 Problem 1 of text book) 15 points

   Since it is a monotone QSAT Boolean formula, if an instance with x_1 to x_n is satisfied, changing one or more its literal from zero to one, leave it satisfied. However, if an instance is not satisfied, changing one or more literals from zero to one can make it satisfied. Thus, if an instance is not satisfiable, changing one literal from zero to one might make it satisfiable, but changing one literal from one to zero will not make it satisfiable.

   Thus, to check the formula is satisfiable, set all literals with odd index equal to 1. This is the case that is most likely to satisfy the instance.

   If the instance is satisfied when even literals are equal to 0, then it is satisfied when even literals are equal to 1.

   Hence, we only need to check if the formula is satisfiable when all odd literals are one and even literals are zero. If the formula is satisfied, then the formula is satisfiable. If not, then the formula is not satisfiable. That is there is not change in odd literals that can make the formula satisfied for all values of even literals.

   In short: set all odd literals equal to one and even literals equal to zero. Iff the formula is satisfied, then the formula is satisfiable. It can be checked in polynomial time.