Exam Combinatorial Optimization, 28 October 2021.

- It is **not** allowed to use any books, notes, or calculator. Just pen and paper.
- You may also obtain points for partial answers.
- Clearly indicate where your final answer is in case you make some computations.

1a	1b	1c	1d	1e	2	3a	3b	4a	4b	4c	5	6	7a	7b	7c	$\sum$
2	2	2	2	2	5	2	4	2	3	4	3	6	1	2	3	45

Question 1 (10 points)
The table shows an instance of the weighted Set Cover problem. A value '1' means that element e_i is included in set S_j. For example, $S_1 = \{e_1, e_2\}$. The second row gives the weights of the sets.

	S_1	S_2	S_3	S_4	S_5	S_6
	21	22	23	31	32	33
e_1	1	1	0	0	0	0
e_2	1	0	1	0	0	0
e_3	0	1	1	0	0	0
e_4	0	0	0	1	1	0
e_5	0	0	0	1	0	1
e_6	0	0	0	0	1	1

(a) (2pt.) Give an optimal solution and its value for this set cover instance.

(b) (2pt.) Give the ILP-formulation for **this** set cover instance.

(c) (2pt.) The LP-relaxation to the LP has a solution of value 81. Give such a solution.
(Hint: $\sum_j w_j = 162$.)

(d) (2pt.) Give the the dual of the LP-relaxation of the formulation given in (b). Use a variable y_i for each element e_i.

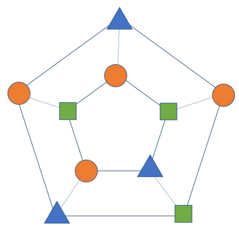
(e) (2pt.) The following is an optimal solution to the dual:
$y_1 = 10, y_2 = 11, y_3 = 12, y_4 = 15, y_5 = 16, y_6 = 17$.
We have seen in Chapter 1 an approximation algorithm for set cover that starts with an optimal solution for the dual. What are the sets chosen by this algorithm when it starts with this dual solution and what is the value (the sum of weights of the chosen sets)?

Question 2 (5 points)
Remember that an *independent set* in a graph $G = (V,E)$ is a subset $S \subseteq V$ of vertices such that no edge $e \in E$ has both endpoints in S.

Consider the following optimization problem:

MAXMIN-INDEPENDENT SET:

Instance:	Graph $G = (V,E)$
Solution:	A partition of the vertices into sets $S_1, S_2, \ldots, S_k \subseteq V$ for some k such that each of the sets S_i $(i = 1,\ldots,k)$ is an independent set in G.
Value:	The size of the smallest set: $\min\{|S_1|, |S_2|, \ldots, |S_k|\}$
Goal:	Maximize value.



The optimal value for the MaxMin-Independent set problem for the graph above is 3. The figure shows a partition in 3 independent sets of size $3,3$, and 4.

Now assume that you are given a *connected* graph on $n = 1000$ vertices and you know that the optimal value for the MaxMin-Independent set problem for that instance is 400. Argue that an optimal *solution* (of value 400) for that instance is easy to find. (If you refer to some known result or algorithm then do explain that as well.)

Question 3 (6 points)

$$C_1 = x_1, \qquad C_2 = x_1 \vee \overline{x_2}, \qquad C_3 = \overline{x_1} \vee \overline{x_3} \qquad C_4 = x_2 \vee x_3$$

(a) (2pt.) Shown is an instance of the (unweighted) maximum satisfiability problem. What is the expected number of satisfied clauses when you assign each variable independently the value 'True' with probability 0.5? Show your computation.

(b) (4pt.) Now suppose that you apply the method of conditional expectations to derandomize this approach. Assume that you go through the variables in the order x_1, x_2, x_3. Then what assignment do you get? Show your computation.

Question 4 (9 points)

For this question, we first recap some theory of the Prize-collecting Steiner Tree problem.

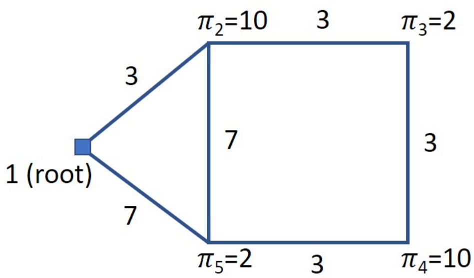
PRIZE-COLLECTING STEINER TREE:

Instance:	$G=(V,E)$ and a cost c_e for every edge $e \in E$ and a penalty π_i for every vertex $i \in V$. Also given is a root $r \in V$.
Solution:	Tree T containing r. Let $V(T)$ be the vertices in T
Value:	$\sum_{e \in T} c_e + \sum_{i \in V-V(T)} \pi_i$.
Goal:	Find a solution of minimum cost.

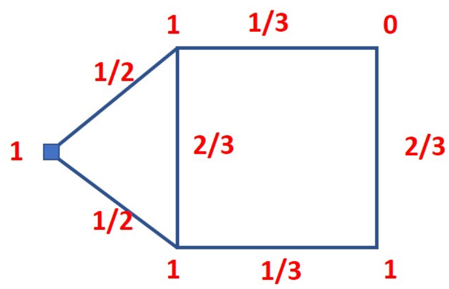
An ILP for the problem is as follows.

$$\begin{aligned}
\text{(ILP)} \quad \min \quad & Z = \sum_{e\in E} c_e x_e + \sum_{i \in V} \pi_i (1-y_i) \\
s.t. \quad & \sum_{e \in \delta(S)} x_e \geqslant y_i, && \text{for all } i,S \text{ with } i \in S \subseteq V-r, \\
& y_r = 1, \\
& x_e \in \{0,1\}, && \text{for all } e \in E, \\
& y_i \in \{0,1\}, && \text{for all } i \in V.
\end{aligned}$$

In the LP-relaxation, the binary constraints are replaced by $0 \le x_e \le 1$ and $0 \le y_i \le 1$.



Instance with edge- and penalty costs



x- and y-values

(a) (2pt.) What is the optimal value for the instance shown above? (Left figure only. The figure on the right is for (b).)

(b) (3pt.) Is the solution shown on the right a feasible solution to the LP-relaxation? Explain your answer.

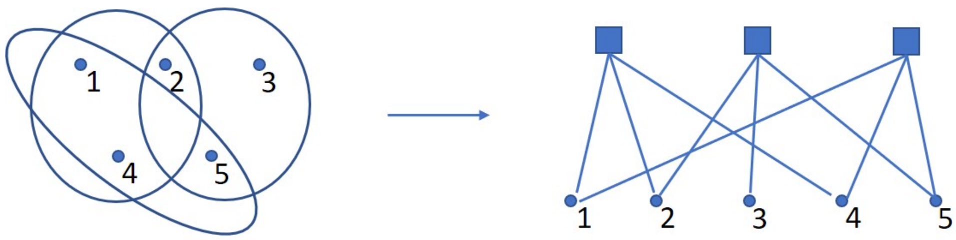
STEINER TREE:

Instance:	$G=(V,E)$ and a cost c_e for every edge $e \in E$ and a set $S \subseteq V$ called the terminals.
Solution:	Tree T containing S.
Value:	$\sum_{e \in T} c_e$.
Goal:	Find a solution of minimum cost.

(c) (4pt.) The (standard) Steiner tree problem is defined as above. Show that any algorithm that solves the Prize-collecting Steiner Tree problem in polynomial time can be used to solve the Steiner Tree problem in polynomial time.

Question 5 (3 points)
Chapter 4 shows a 4-approximation algorithm for the uncapacitated facility location problem (UFL). There, it is assumed that the distances satisfy the triangle inequality. The problem becomes much harder to solve if the triangle inequality is not assumed. In fact, the set cover problem can be modeled as an uncapacitated facility location problem in that case.

The figure shows an example of this reduction. Left is an instance of the set cover problem (3 sets and 5 elements) and right an instance of UFL. There is a facility i for each subset S_i. There is a client j for each element j. Further, there is an edge if the corresponding element is in the subset. The figure does not show how the costs are chosen.



In this reduction from unweighted Set Cover to UFL, how should you define the costs? (Fill in the blanks on your paper. It suffices to write $A = ..., B = ..., C = ...$)
N.B. Your answer should apply in general. Not just for this example. There are more ways to do this but just fill in some numbers that would make this reduction work in general.

(A) For each facility i, the opening cost f_i is ......

(B) If $j \in S_i$ then the connection cost c_{ij} is ......

(C) If $j \notin S_i$ then the connection cost c_{ij} is ......

Question 6 (6 points)
The list scheduling algorithm is known to be a 2-approximation algorithm for minimizing the length of the schedule ('makespan') on identical parallel machines. Assume now that all jobs are relatively small. Precisely, assume that for each job k, its processing time p_k satisfies

$$p_k \leq \frac{1}{3(m-1)} \sum_{j=1}^{n} p_j,$$

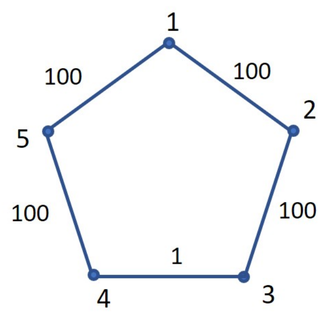
where m is the number of machines and n is the number of jobs. Prove that in this case, list scheduling is a $4/3$-approximation algorithm.

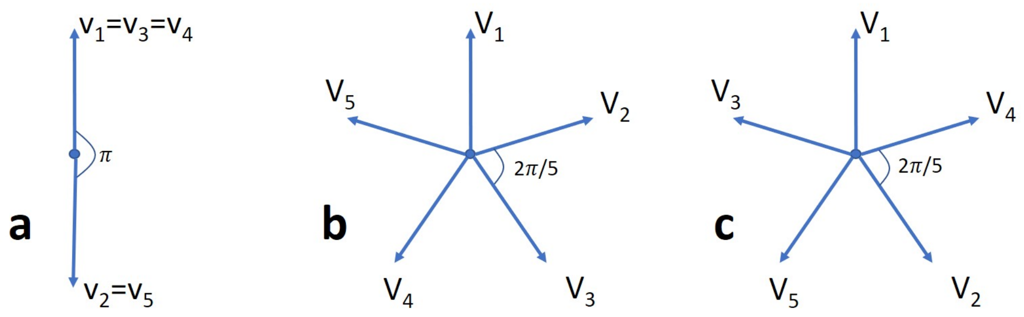
Question 7 (6 points)

Shown below are the Quadratic Program and the Vector Program relaxation for the maximum weighted cut problem, where an instance is an edge-weighted graph $G = (V, E)$ on n vertices.

$$\text{(QP)} \quad \max \quad \frac{1}{2} \sum_{(i,j)\in E} (1 - y_i y_j) w_{ij}$$
$$s.t. \quad y_i \in \{-1, 1\} \qquad i = 1, \dots, n.$$

$$\text{(VP)} \quad \max \quad \frac{1}{2} \sum_{(i,j)\in E} (1 - v_i \cdot v_j) w_{ij}$$
$$s.t. \quad v_i \cdot v_i = 1, \ v_i \in \mathbb{R}^n \qquad i = 1, \dots, n.$$





(a) (1pt.) What is the value of the maximum weighted cut in the graph above? (See top figure.)

(b) (2pt.) Below the graph are 3 feasible solutions to the VP for this instance. Rank the solutions from largest to smallest VP-value. (You do not need to give the value. Just answer 'a, b, c' for example.)

(c) (3pt.) Remember that the max-cut algorithm of Goemans and Williamson (of Chapter 6) was defined as follows: Solve the VP and then apply randomized rounding. Now suppose that instead of the *optimal* VP-solution you take each of the 3 solutions above. For each of the 3 solutions give the the expected weight of the cut.