Exercise 2: Integer Programming (2.5 pts).

1. **(1.5 pts)** A valid solution is the following.

   $$x_j + p_j - x_i \leq M \times y_{i,j} \text{ for all } i \neq j \text{ s.t. } 1 \leq i, j \leq n$$
   $$x_i + p_i - x_j \leq M \times (1 - y_{i,j}) \text{ for all } i \neq j \text{ s.t. } 1 \leq i, j \leq n$$

   where $M = 2 \times \sum_{i=1}^{n} p_i$ is a big enough value so that the first inequation would trivially hold when $y_{i,j} = 1$ and

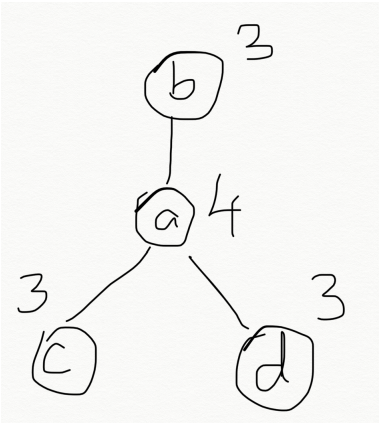
   where $y_{i,j} \in \{0, 1\}$ for all i, j s.t. $1 \leq i, j \leq n$.

   An implementation in CPLEX based on the above solution would be as follows.

```
int n = ...; // number of jobs
range I = 1..n;
int p[i in I] = ...; // hours per job
float w[i in I] = ...; // job weights

dvar int+ x[i in I]; // start time per job
dvar boolean y[i in I,j in I]; // job ordering

int M = 2*sum(i in I) p[i]; // maximum starting time

minimize sum(i in I) w[i]*x[i];

subject to {
       forall(i in I,j in I: i != j) {
           x[j] + p[j] - x[i] <= M*y[i,j];
           x[i] + p[i] - x[j] <= M*(1-y[i,j]);
       }
}
```

2. **(1 pt)** The given algorithm is not a 2-approximation for the Minimum Weight Vertex Cover problem.

A counterexample is the star S_3 with the following weights:



for which the above algorithm would select the vertices b, c, and d with a total weight of 9 instead of the vertex a (which also constitutes a vertex cover) with weight 4. Since $9 > 2 \times 4$, the solution is not a 2-approximation.