EXERCISE ONE Consider the classic NP-hard KNAPSACK PROBLEM, where we have $n$ objects $a_1, \ldots, a_n$, each object has a weight $w_i$ and cost $c_i$, and our bag has a weight limit of $B$.

Find a greedy 2-approximation algorithm for this problem.

EXERCISE TWO Consider SCHEDULING WITH DEPENDENCIES: we schedule jobs of different lengths on $m$ computers ($m$ is a part of the input), but we also have a *dependence graph* on the jobs, and we can schedule a job only when all its dependencies are completed.

Find a greedy 2-approximation algorithm for this problem.

---

EXERCISE THREE You may recall MAX SAT from the last exercise session, where we formulated a randomized approximation algorithm for it. This algorithm was effective for clauses of length 2 or more, but when there were too many clauses of type $(x_i)$ or $(\neg x_j)$, it was only a 1/2-approximation.

Let us prove that we can assume the input is a little bit nicer:

1. Prove the following: Suppose we have a $c$-approximation algorithm for a subset of MAX SAT – it only works on inputs which contain no negative mono-clauses like $(\neg x_i)$. Then we can transform it into a $c$-approximation algorithm for MAX SAT on all inputs.
2. Prove that the same holds for WEIGHTED MAX SAT, where each clause has weight $w_i$ and we maximize the weighted sum of satisfied clauses, i.e. $\max \sum_i w_i C_i$.

EXERCISE FOUR We have learned from the previous exercise that we can only deal with MAX SAT on inputs that contain no negative mono-clauses like $(\neg x_i)$. We should use this fact to choose a better probability $p$, which we use in the randomized algorithm for setting a variable to 1:

1. Prove that if all variables $x_i$ are randomly set to be true with probability $p > \frac{1}{2}$, then the probability of satisfying a clause is at least $\min(p, 1-p^2)$.
2. Choose a good $p$ and finish the analysis of the suggested randomized algorithm for MAX SAT.

---

EXERCISE FIVE We now consider MAX DICUT. On the input we get a directed graph $G = (V, \vec{E})$ and a non-negative weight function on the edges. Our task is to find a subset of vertices $S$ so that $\vec{E}(S, V \setminus S)$ (the edges directed from $S$ to the rest) have maximum possible weight.

Suggest a probabilistic $\frac{1}{4}$-approximation algorithm for MAX DICUT.

EXERCISE SIX Let us try to improve on our algorithm for MAX DICUT:

1. Suggest a natural $\{0,1\}$-integer program solving MAX DICUT.
2. Choose each vertex $v_i$ with probability $1/4 + x_i^*/2$, where $x_i^*$ is the optimum of the linear relaxation of the previous integer program. Show that it is a 1/2-approximation.