Info: Whenever it is asked to implement something with python-mip you can also use any other similar library and/or programming language of your choice. Just shortly ask in Discrod to clarify if this is fine for me.
Also, if not specified in detail, for all programming exercises generate some interesting test cases to run the program and output solutions and objective values in a reasonable format.

1 Robust Knapsack MIP

1.1 Knapsack MIP

Implement a MIP formulation of the Knapsack Problem using python-mip.

1.2 Knapsack MIP: Interval uncertainties

Implement the MIN-MAX robust version of the Knapsack problem using python-mip.

1.3 Knapsack MIP: Discrete uncertainties

Formulate the MIN-MAX robust knapsack problem as a mixed integer program. Implement this formulation using python-mip.

2 Min-Max-Regret Spanning Tree

2.1 Characterization of strong edges

Prove the following theorem from the lecture:
Let $e \in E$. Then e is a strong edge, iff there exists a spanning tree T such that $e \in T$ and T is a minimum spanning tree with respect to the cost function $w^{\text{wc}(e)}$ defined as

$$c^{\text{wc}(e)}(e') = \begin{cases} c^+(e') & \text{if } e = e' \\ c^-(e') & \text{if } e \neq e'. \end{cases}$$

2.2 Implementation of the MIP formulation

Implement the MIP formulation from the lecture using python-mip (or some other library for MIP formulations).

2.3 Finding weak and strong edges

Explain how to modify Kruskals algorithm to efficiently compute the set of weak and strong edges

3 Min-Max-Regret Shortest Path

3.1 MIP formulation

Obtain a MIP formulation for the MIN-MAX Regret Shortest Path problem with interval uncertainties similar to the formulation ofr the min-max regret spanning tree problem from the lecture.

3.2 Preprocessing theory (hard exercise)

Can you adapt some of the ideas for weak and strong edges to the min-max regret shortest path problem? Also, try to show how they can be computed efficiently!

3.3 Preprocessing implementation (hard exercise)

Implement the preprocessing and compare the performance with directly running the MIP model stated in 3.1.