Discrete Optimization
Task 2 - Knapsack

Student: Horace Guy Report on Task 2, MA2827 2017

I worked in collaboration with Olivier on this task. In regard of the algorithms complexity we saw in class, we decided to implement two techniques : Dynamic programming was perfectly suited for the first two tests, whereas a branch-and-bound approach is necessary to solve the last three.

The file solver.py implements a `Solver` class to program conveniently.

The dynamic programming function reaches RecursionError for the tests 3 to 5.

Branch and bound was implemented with depth-first search, recursive and then iterative version, and limited discrepancy search. The iterative technique is necessary to solve the last test. LDS is not necessary but finds solution 5 with 1 error very quickly. All of these implementations use the same heuristic: fractional items.

In order to accelerate the depth-first search, I sorted the items by density. I used the `numpy` module to perform `argsort` and dense sort according to Dantzig's heuristic. The `deque` data structure is useful to implement the iterative version of the depth-first search in branch and bound. Finally, the dynamic programming program uses the `lru_cache` decorator to conviently store the computed values in cache.