Permutation scheduling:

The following files are to be included in a single zip archive:

1. Source code solving the scheduling problem. Use an appropriate programming language or a computational environment.
2. Input data file, see the following description.
3. A file including results according to following guidelines:
   a. A list with the starting schedule.
   b. A list with durations of all tasks p_j.
   c. A list with deadlines d_j.
   d. A list with weights of each task task w_j.
   e. Find the best schedule and its price for each step.
   f. The best schedule, including its price.

All tasks are based on the following specification of a scheduling problem:

1. Permutation scheduling and a single machine $1|d_j|\sum w_j T_j$
2. Find the optimal schedule using tabu search

We suggest that you create a smaller instance of the problem and use it verify that your program behaves correctly.

Scheduling problem:

- Optimization criterion: minimize the accumulated non-negative delay.
- Schedule neighborhood: All schedules acquired by a pair-wise swap of neighboring tasks.
- Schedule selection from neighborhood: Finds the best schedule for each step from the neighborhood.
- Tabu list: pairs of tasks, swapped in last 11 modifications.
- There are 15 tasks with names from “1” to “15”.
- The starting schedule is `[2, 3, 15, 6, 5, 10, 8, 7, 13, 14, 1, 4, 12, 9, 11]`.
- Perform 200 iterations.
- Task lengths are `(4, 16, 6, 5, 9, 19, 1, 13, 12, 20, 20, 19, 5, 18, 12)`.
- Deadlines are `(36, 67, 105, 53, 77, 124, 194, 157, 25, 202, 43, 61, 5, 7, 8)`.
- Weights are `(21, 45, 35, 73, 1, 28, 21, 14, 76, 70, 51, 23, 69, 62, 80)`.