Max weight Interval Pack ing

- Intervals \( I_i = (s_i, t_i) \) of real line
- two intervals conflict if \( I_i \cap I_j \neq \emptyset \)
- Each interval has weight \( w_i \geq 0 \)

Pick max weight collection of intervals that don't conflict:

1. if \( w_i = 1 \) \( \forall i \Rightarrow \) greedy algo works (HWO).
2. else it can fail (example! try to construct!)

Solution here: DP

Some notation:
- say sort the intervals by their end time \( t_j \)
- so that \( t_1 \leq t_2 \leq \ldots \leq t_n \).

For \( I_j \), let \( p(j) \) be index of "previous" interval in list that does not conflict

\[ p(j) = \max \{ i: t_i < t_j \text{ and } I_i, I_j \text{ do not conflict} \} \]

if all conflict then \( p(j) = 0 \).

\[ p(1) = 0 \]
\[ p(2) = 0 \]
\[ p(3) = 1 \]

Q: how fast can you compute these \( p(j) \) values?

\( O(n \log n) \) possible.
\[ \text{OPT}(j) = \begin{cases} \text{weight of optimal sol on sub-instance of intervals } I_1, \ldots, I_j \\ \text{OPT}(0) = 0, \\ \text{OPT}(j) = \max \{ \text{OPT}(j-1), \ W_j + \text{OPT}(p(j)) \} \end{cases} \]

takes exponential time

So, introduce tables (memoization, or just bottom-up table filling)

\[
\text{MWIP}(j) \begin{cases} \text{if } j = 0 \text{ return 0} \\ \text{if } \text{Memo}(j) \text{ undefined return } \text{Memo}(j) \\ \max \{ \text{MWIP}(j-1), \ W_j + \text{MWIP}(p(j)) \} \end{cases} \]

\]

\[
\text{return Memo}(j) \]

And the actual program calls:

\[
\text{main} \{ \\ \text{MWIP}(n) \} \]
New given memo table, get actual solution (not just weight) as follows:

Actual solution is

```c
Print Sol(i)
```

```c
if Memo(j-1) ≥ Memo(p(j)) + wj
    Print Sol(j-1)
else
    Print Sol(p(j)); print(*j).
```

This basically goes over the solution, and reconstructs the choices we made.