QUESTION: SUPPOSE THAT V IS AN ARRAY OF N ELEMENTS; $V[i]$ IS THE SCORE THAT STUDENT i GOT IN THE EXAM.
SUPPOSE, FURTHER, THAT THERE ARE THREE POSSIBLE SCORES ("INSUFFICIENT": 0, "PASS": 1, "HONORS": 2).

SORT V AS FAST AS POSSIBLE.
(IN PARTICULAR, BEAT $O(N \log N)$).

$V = [2,1,1,0,1,1,0,2] \Rightarrow [0,0,1,1,1,1,2,2]$

$W = [0, 0, \dots, 0]$

HEAP-SORT WOULD TAKE $\Theta(N \log N)$ $\left(\equiv \begin{array}{l} O(N \log N) \\ \Omega(N \log N) \end{array}\right)$

```
DEF cSort(V):
    A = [0, 0, 0]
    FOR x IN V:
        A[x] += 1
    W = ([0] * A[0]) + ([1] * A[1]) + ([2] * A[2])
    RETURN W
```

$O(n)$

$O(t \cdot n)$

- LOCAL SEARCH ALGORITHM
- GREEDY ALG.
- DYNAMIC PROGRAMMING

WEIGHTED INTERVAL SCHEDULING

WE ARE GIVEN A SET OF INTERVALS

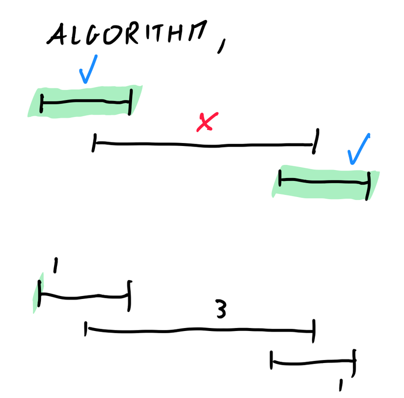
$$I = \{I_1, \dots, I_n\} = \{(s_1, f_1), \dots, (s_n, f_n)\}$$

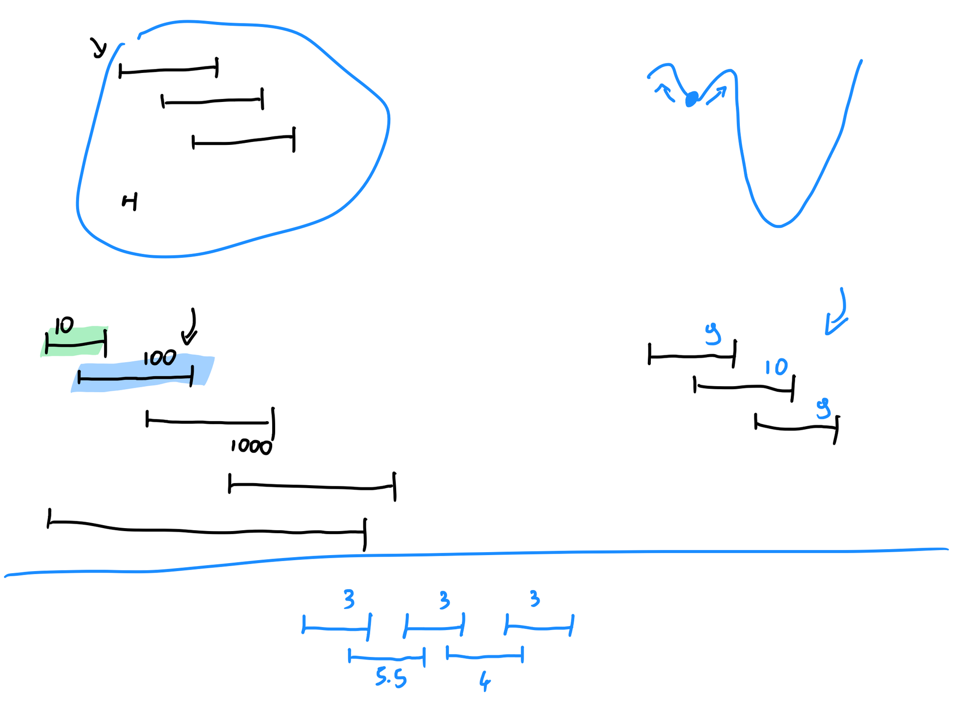
(WHERE s_j IS THE STARTING TIME OF INTERVAL I_j, AND f_j IS ITS FINISHING TIME), AS WELL AS A WEIGHTING OF THE INTERVALS $w(I_j) = w_j \geq 0, \forall j \in \{1, \dots, n\}$.

GOAL: FIND A SUBSET $S \subseteq I$ OF THE INTERVALS, THAT IS NON-OVERLAPPING, AND THAT HAS MAXIMUM WEIGHT/VALUE $\sum_{I_j \in S} w_j$.

INTERVAL SCHEDULING IS A SPECIAL CASE OF WEIGHTED INTERVAL SCHEDULING.
(AND WIS IS A GENERALIZATION OF IS)

WE SOLVED IS WITH "EARLIEST-FINISHING-TIME" GREEDY ALGORITHM,





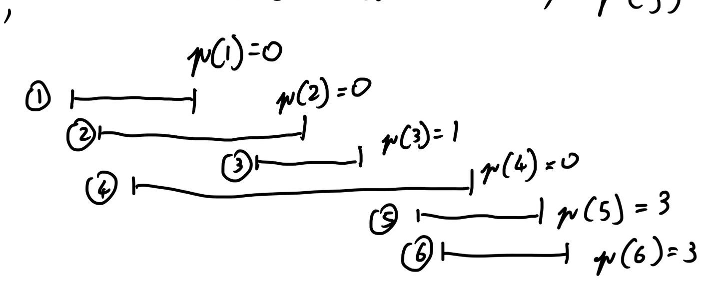
"DYNAMIC PROGRAMMING"

AN ALGORITHMIC TECHNIQUE THAT KEEPS TRACK OF ALL THE SOLUTIONS AT ONCE, IN AN "EFFICIENT" MANNER.
(GREEDY ALGORITHMS, INSTEAD, KEEP TRACK OF A SINGLE SOLUTION AT EACH TIME).

FIRST OF ALL, LET US SORT THE INTERVALS BY FINISHING TIME: $f_1 \leq f_2 \leq \dots \leq f_n$.

DEF: INTERVAL i "COMES BEFORE" INTERVAL j IFF $f_i \leq f_j$.

DEF: LET $p(j)$, FOR AN INTERVAL j, BE THE LARGEST INDEX $i < j$ S.T. INTERVALS i AND j ARE DISJOINT ("COMPATIBLE", "NON-OVERLAPPING"), OR, IF NO SUCH i EXISTS, $p(j) = 0$.



L: THE GENERIC INTERVAL i IS DISJOINT (COMPATIBLE) WITH EACH OF THE INTERVALS $1, 2, \dots, p(i)$, AND IT IS NOT DISJOINT FROM INTERVAL $p(i)+1$.

P: EXERCISE.

SUPPOSE THAT O_i IS AN OPTIMAL SOLUTION OF THE PROBLEM RESTRICTED TO THE INTERVALS $\{I_1, I_2, \dots I_i\} = \{(s_1, f_1), (s_2, f_2), \dots, (s_i, f_i)\}$.

LET OPT_i BE THE VALUE OF O_i.
$\left(OPT_i = \sum_{j \in O_i} w_j\right)$

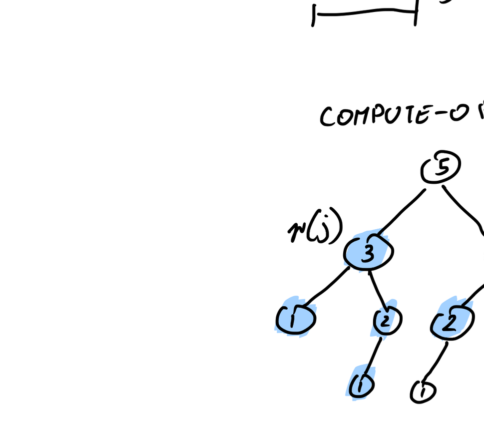
OBS: $\forall j \geq 1$, (i) $w_j + OPT_{p(j)} \geq OPT_{j-1}$ IFF ("IF AND ONLY IF") THERE EXISTS AN OPTIMAL SOLUTION O_j SUCH THAT $I_j \in O_j$,
(ii) $OPT_{j-1} \geq w_j + OPT_{p(j)}$ IFF THERE EXISTS AN OPTIMAL SOLUTION O_j SUCH THAT $I_j \notin O_j$, THUS
(iii) $OPT_j = \max(w_j + OPT_{p(j)}, OPT_{j-1})$.

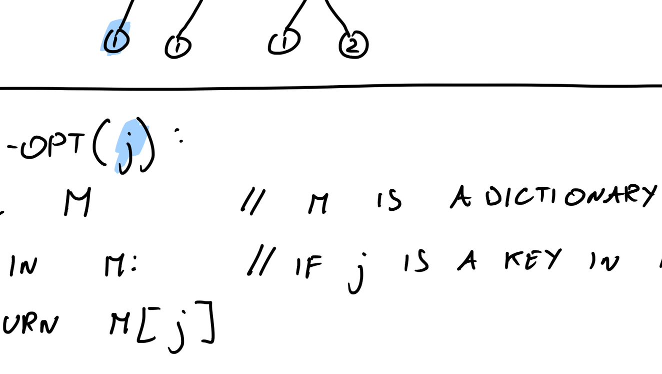
P: A SOLUTION O_{j-1} TO THE $(j-1)$-PROBLEM (THE PROBLEM RESTRICTED TO THE FIRST $j-1$ INTERVALS), ACTS ALSO AS A SOLUTION (WITH THE SAME VALUE) TO THE j-PROBLEM.
THUS, $OPT_j \geq OPT_{j-1}$.
MOREOVER, A SOLUTION $O_{p(j)}$ TO THE $p(j)$-PROBLEM CAN BE TRANSFORMED INTO THE SOLUTION $O_{p(j)} \cup \{I_j\}$ (WITH VALUE $OPT_{p(j)} + w_j$) TO THE j-PROBLEM.
THUS, $OPT_j \geq OPT_{p(j)} + w_j$ AND $OPT_j \geq \max(OPT_{p(j)} + w_j, OPT_{j-1})$.
WE NOW PROVE THAT $OPT_j = \max(OPT_{p(j)} + w_j, OPT_{j-1})$.
LET O_j BE AN OPTIMAL SOLUTION TO THE j-PROBLEM:
- IF $I_j \notin O_j$, THEN O_j IS ALSO AN OPTIMAL SOLUTION (WITH THE SAME VALUE) TO THE $(j-1)$-PROBLEM. THUS, IF $I_j \notin O_j$, $OPT_j = OPT_{j-1}$.
- IF $I_j \in O_j$, THEN $O_j - \{I_j\}$ IS AN OPTIMAL SOLUTION (OF VALUE $OPT_j - w_j$) TO THE $p(j)$-PROBLEM. THUS, IF $I_j \in O_j$, $OPT_j = OPT_{p(j)} + w_j$.

THUS, (i), (ii) AND (iii) FOLLOW.

```
DEF COMPUTE-OPT(j):
    IF j == 0:
        RETURN 0
    ELSE:
        RETURN max(w_j + COMPUTE-OPT(p(j)), COMPUTE-OPT(j-1))
```

L: COMPUTE-OPT(j) RETURNS OPT_j, FOR EACH $j = 0, 1, \dots, n$.





```
M-COMPUTE-OPT(j):
    GLOBAL M          // M IS A DICTIONARY
    IF j IN M:        // IF j IS A KEY IN M
        RETURN M[j]
    ELSE:
        IF j == 0:
            M[j] = 0
        ELSE:
            M[j] = max(w_j + M-COMPUTE-OPT(p(j)), M-COMPUTE-OPT(j-1))
    RETURN M[j]
```

L: M-COMPUTE-OPT(j) RETURN OPT_j.

L: M-COMPUTE-OPT(j) TAKES $O(j)$ TIME.