HW Solution #7

Only P1-P5 will be graded.

P1:

There could be different cutting methods, but the revenue shall be the same.

(1) revenue 16, cuts: [2, 6]

(2) revenue 214, cuts: [6, 12, 12]

P2: Table 1 shows a counterexample.

length i	1	2	3	4
price p_i	1	20	33	36
p_i/i	1	10	11	9

Table 1: Problem 2

P3: No. Because the mergesort algorithm splits the array and the subproblems do not overlaps.

P4: No. Suppose the rod was length 4, the values were $l_1 = 2, l_2 = l_3 = l_4 = 1$, and each piece has the same worth regardless of length. Then, if we make our first cut in the middle, we have that the optimal solution for the two rods left over is to cut it in the middle, which isn't allowed because it increases the total number of rods of length 1 to be too large.

P5: The problem is similar to knapsack problem and the subproblem is whether to choose each job.

We can first sort the jobs in the order of increasing finish time. For a job j, we can further define $p(j)$ = largest index $i < j$ such that job i is compatible with j.

Use $OPT(j)$ to represent the value of optimal solution to the problem consisting of job $1, 2, ..., j$. Every time when deciding whether to include the job, we looks at

1. Case 1 to include job j: We cannot use the incompatible jobs $\{p(j)+1, p(j)+2, ..., j-1\}$. We can use the optimal solution from $p(j)$ where consisting the remaining compatible jobs. Note that as the o $OPT(p(j))$ is already optimal for the subproblem. We don't need to further checks the jobs before $p(j)$.
2. Case 2 to exclude job j: then the value is the same as $OPT(j-1)$.

The optimal substructure can be written as

$$OPT(j) = \begin{cases} 0, & j = 0 \\ \max\{v_j + OPT(p(j)), OPT(j-1)\}, & otherwise \end{cases}$$

You can find source codes to solve the problems here.