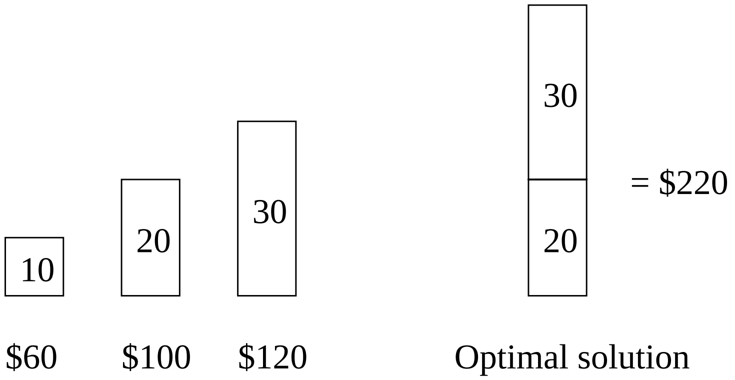
CPS 130 Homework 15
Dynamic Programming

Write and justify your answers in the space provided.[1]

1. In this problem we consider the 0-1 KNAPSACK PROBLEM: Given n items, with item i being worth $v[i]$ dollars and having weight $w[i]$ pounds, fill a knapsack of capacity m pounds with the maximal possible value.

Example: Given a knapsack of capacity 50, the maximal value obtainable with three items of value \$60, \$100, and \$120 and weights 10, 20, and 30, respectively, is \$220.



The algorithm `Knapsack(i,j)` below returns the maximal value obtainable when filling a knapsack of capacity j using items among items 1 through i (`Knapsack(n,m)` solves our problem). The algorithm works by recursively computing the best solution obtainable *with* the last item and the best solution obtainable *without* the last item, and returning the best of them.

```
Knapsack(i,j)

  IF w[i] <= j THEN
    with = v[i] + Knapsack(i-1, j-w[i])
  ELSE
    with = 0
  END IF
  without = Knapsack(i-1,j)
  RETURN max{with, without}

END Knapsack
```

[1] Collaboration is allowed, even encouraged, provided that the names of the collaborators are listed along with the solutions. Students must write up the solutions on their own.

(a) Show that the running time T of `Knapsack`(n, m) is exponential in n or m. (*Hint:* look at the case where $w[i] = 1$ for all $1 \leq i \leq n$ and show that $T(n, m) = \Omega(2^{\min(m,n)})$).

(b) Describe an $O(n \cdot m)$ algorithm for computing the value of the optimal solution.