Problem #1

Problem 1.

A set S of 5 items with weights $w_i > 0$ and dollar values $d_i > 0$ for $i = 1, 2, 3, 4, 5$

$S = \{(item_1, 10, 100), (item_2, 20, 160), (item_3, 30, 180), (item_4, 40, 160), (item_5, 50, 100)\}$

(30 → weights, 180 → dollar)

$W = 50$ (Weight of the knapsack)

S is sorted based on decreasing benefits/weight ratios.

a)

	benefits/weights	weights	benefits
item 1	$ 10	10	$ 100
item 2	$ 8	20	$ 160
item 3	$ 6	30	$ 180

weight: $10 + 20 + 20 = 50$

benefits: $\$100 + \$160 + 30 \times \frac{2}{3} \times 6 \; (= \$120) = 380$

∴ Optimal benefit: $380

∴ Solution: items 1, 2, and $\frac{2}{3}$ of item 3.

The greedy approach works for fractional knapsack.

b) Greedy Solution: (b/w=10, b/w=8)

weight ⇒ $10 + 20 = 30$ (item 1 + item 2)

benefit ⇒ $100 + 160 = \$260$

Optimal solution: (b/w=8, b/w=6)

weight ⇒ $20 + 30 = 50$ (item 2 + item 3)

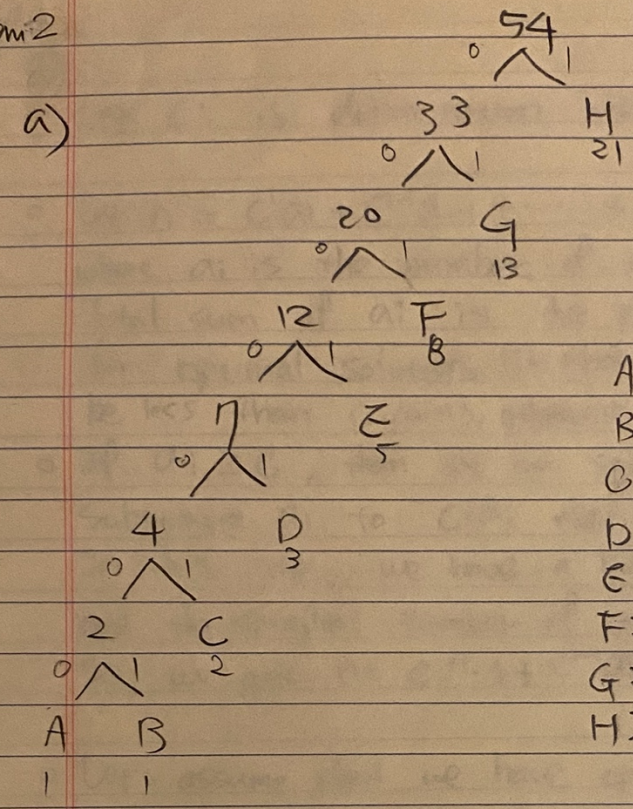
benefits ⇒ $160 + 180 = \$340$

∴ The greedy approach fails for 0-1 knapsack.

Problem #2

Problem 2

a)



A: 0 0 0 0 0 0 0 0
B: 0 0 0 0 0 0 0 1
C: 0 0 0 0 0 0 1
D: 0 0 0 0 0 1
E: 0 0 0 0 1
F: 0 0 1
G: 0 1
H: 1

b) E F D A F F

0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1

E F D A F F

Problem #3

Problem #3

a)

Let c^i is denomination where $0 \le i \le k$, $k \ge 1$, $C > 1$.

- Let $n = c^i a_i + c^{i-1} a_{i-1} + \cdots + c^0 a_0$ where a_i is the number of each denomination c_i. Total sum of a_i is the number of coins for optimal solution. We should assume that a_i must be less than C coins, otherwise number of coins will be increased.
- If $a_i \ge C$, then we can say $a_i = C$. Substitute a_i to $c_i a_i$ then we get $C \cdot C^i = C^{i+1}$. In this case, we have a bigger denomination of C. and the smaller number of coins, a_{i+1} of value 1. So, we get $n = c^{i+1} \cdot 1 + c^{i-1} a_{i-1} + \cdots + c^0 a_0$.

- Let's assume that we have optimal solution for m cents. d^i is the denomination where $0 \le i \le k, k \ge 1, d > 1$. $m = d^i O_i + d^{i-1} O_{i-1} + \cdots + d^0 O_0$ where O_i is the number of each denomination d_i. So we have $O_i + O_{i-1} + O_{i-2} + \cdots + O_0$ of coins for optimal solution. If $O_i \ge d$, then $m = d^{i+1} + d^{i-1} O_{i-1} + \cdots + d^0 O_0$, and we have $1 + O_{i-1} + O_{i-2} + \cdots + O_0$ of coins for optimal solution which is less than the assumption, $O_i + O_{i-1} + O_{i-2} + \cdots + O_0$ ($O_i + O_{i-1} + O_{i-2} + \cdots + O_0 > 1 + O_{i-1} + O_{i-2} + \cdots + O_0$ because $1 < d \le O_i$) Having a less coins for m cents is a contradiction to the assumption of optimality.

- We can conclude that optimal solution, picking the largest denomination (C^k) coin less than C coins first, will lead to globally optimal (greedy) choice.
- And, making the greedy choice at each step produces an optimal solution inductively. (Optimal Substructure)

b) To design O(nk) algorithm, we can use dynamic programming

```
coinChange(C, k, n)
    denomi = [C^0, C^1, C^2, ..., C^k]
    M = two dimensional array of n columns
        and k+1 rows, and initialize each cell to 0.
    for j from 0 to n        // Initialize pennies.
        M[0][j] = j

O(nk) [ for i from 1 to k
          for j from 1 to n
            if denomi[i] > j
              M[i][j] = M[i-1][j]
            else
              M[i][j] = minimum(1 + M[i][j-denomi[i]],
                                M[i-1][j])
    return M
```

This algorithm fills every cell in a $k \times n$ matrix. So, time efficiency is $O(nk)$.

Problem #5

Problem#5 If we approach this problem to identify the specific pattern of frequencies, fibonacci recurrence is a good algorithm to generalize the pattern.

- 1 1 2 3 5 8 13 21

$f(0)=1$

$f(1)=1$

$f(2)=f(0)+f(1)=2$

$f(3)=f(2)+f(1)=3$

$f(4)=f(3)+f(2)=5$

$\vdots$

$f(n)=f(n-1)+f(n-2)$ $\Rightarrow$ fibonacci pattern.

- To construct huffman code, let n is a length of the list. (e.g. n is 8 in this case)

$A=(n-2)\times'0'+'0'$ $=0000000$

$B=(n-2)\times'0'+'1'$ $=000001$

$C=(n-3)\times'0'+'1'$ $=00001$

$\vdots$

$H=(n-8)\times'0'+'1'$ $=1$

we can say code $=(n-i)\times'0'+'1'$, $2\le i\le n$.

and $(n-i)\times'0'+'0'$ if $i=2$.