Subject _____ Date _____

Problem 2: (Fibonanci)

1. matrix exponentiation computers.

$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n$ * in $O(\log n)$ multiplication using binary exponentiation Yeilding f_n directly.

2. The recurrence $T(n) = T(n/2) + O(1)$ fits case 2 of the master theorem : $T(n) = O(\log n)$. The decomposition halves n each step, requiring only $O(\log n)$ matrix multiplications, hence time complexity $\Theta(\log n)$:

Subject Date

Problem 2 (Levenshtein (Edit) Distance)

Edit distance between "Saturday" and "Sunday"

= 3

operations:

SATURDAY

SUNDAY

changes: A→U, T→N, insert R?

wait, check:

Alignment:

SATURDAY

SUNDAY__

Substitute: A→U(2), substitute T→N(2)

R(3), align A, align Y. Delete

so 3 edits

Subject _____ Date _____

Problem 3 (0/1 knapsack Algorithm)

1. Greedy fail because choosing highest value / weight ratio may not ~~yield optimal~~ yield optimal solution, due to integer constraints and leftover capacity; dynamic programming garantiee, optimal by considering all subsets.

Solve

2. Solve for course Ex (weights and value) : Dp table of size $(n+1) * (w+1)$.
Fill using :

$$dp[i][w] = \max(dp[i-1][w], dp[i-1][w-w_i]) + w_i$$

3. Yes, space complexity reduced to $O(w)$ by using a 1D array updated right-to-left for each time.