Design & Analysis of Algorithm (Lab)

Name: Ananya

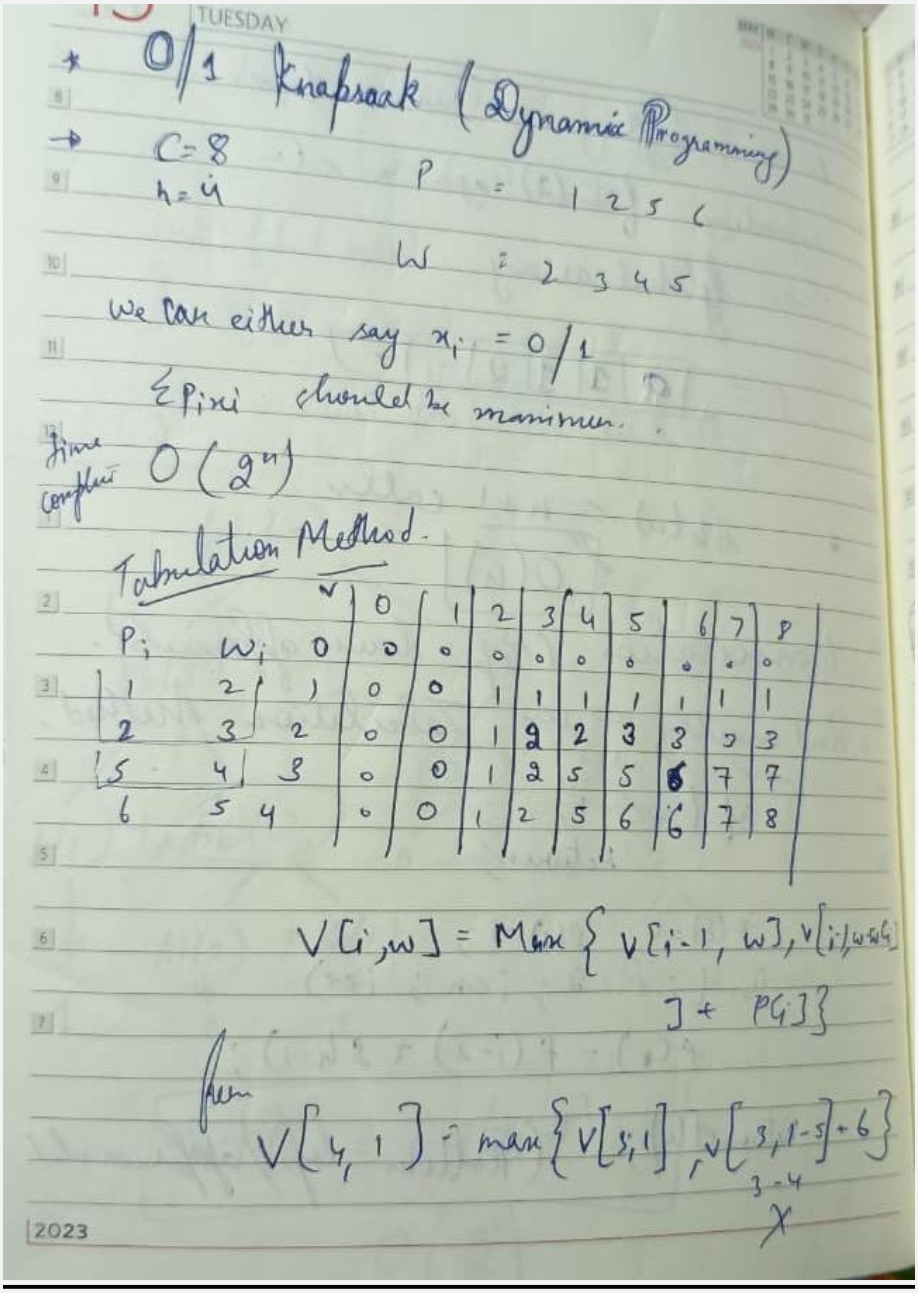
SAPID: 590013832

B-33

Submitted to: Mr.Aryan Gupta

https://github.com/ananya438/DAALAB_ANANYA-590013832

0/1 knapsack (DP)



TUESDAY

0/1 Knapsack (Dynamic Programming)

C = 8

n = 4

P = 1 2 5 6

W = 2 3 4 5

We can either say $x_i = 0/1$

$\sum P_i x_i$ should be maximum.

Time complexity $O(2^n)$

Tabulation Method.

P_i	W_i	v \ w	0	1	2	3	4	5	6	7	8
		0	0	0	0	0	0	0	0	0	0
1	2	1	0	0	1	1	1	1	1	1	1
2	3	2	0	0	1	2	2	3	3	3	3
5	4	3	0	0	1	2	5	5	6	7	7
6	5	4	0	0	1	2	5	6	6	7	8

$V[i,w] = \max\{ V[i-1, w], V[i, w-w_i] + P_i \}$

Then

$V[4,1] = \max\{ V[3,1], V[3,1-5] + 6 \}$

$3-4$

X

2023

```
public class Knapsack {

    static int knapsack(int W, int wt[], int val[], int n) {

        int dp[][] = new int[n + 1][W + 1];

        for (int i = 0; i <= n; i++) {
            for (int w = 0; w <= W; w++) {
                if (i == 0 || w == 0)
                    dp[i][w] = 0;
                else if (wt[i - 1] <= w)
                    dp[i][w] = Math.max(val[i - 1] + dp[i - 1][w - wt[i - 1]], dp[i - 1][w]);
                else
                    dp[i][w] = dp[i - 1][w];    }
        }

        return dp[n][W];
    }
```

O/P:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\nannu\Desktop\JAVA DSA\DAA> cd "c:\Users\nannu\Desktop\JAVA DSA\DAA\0\" ; if ($?) { javac Knapsac
Maximum Profit = 220
PS C:\Users\nannu\Desktop\JAVA DSA\DAA\0>
```

Complexity Analysis (Dynamic Programming)

Time Complexity

O(n × W) → we fill a table of size n * W.

Space Complexity

O(n × W) → because of the 2D DP array.