CPSC 335

Handout 4 (Dynamic Programming) Name Nicholas Ayson

==========

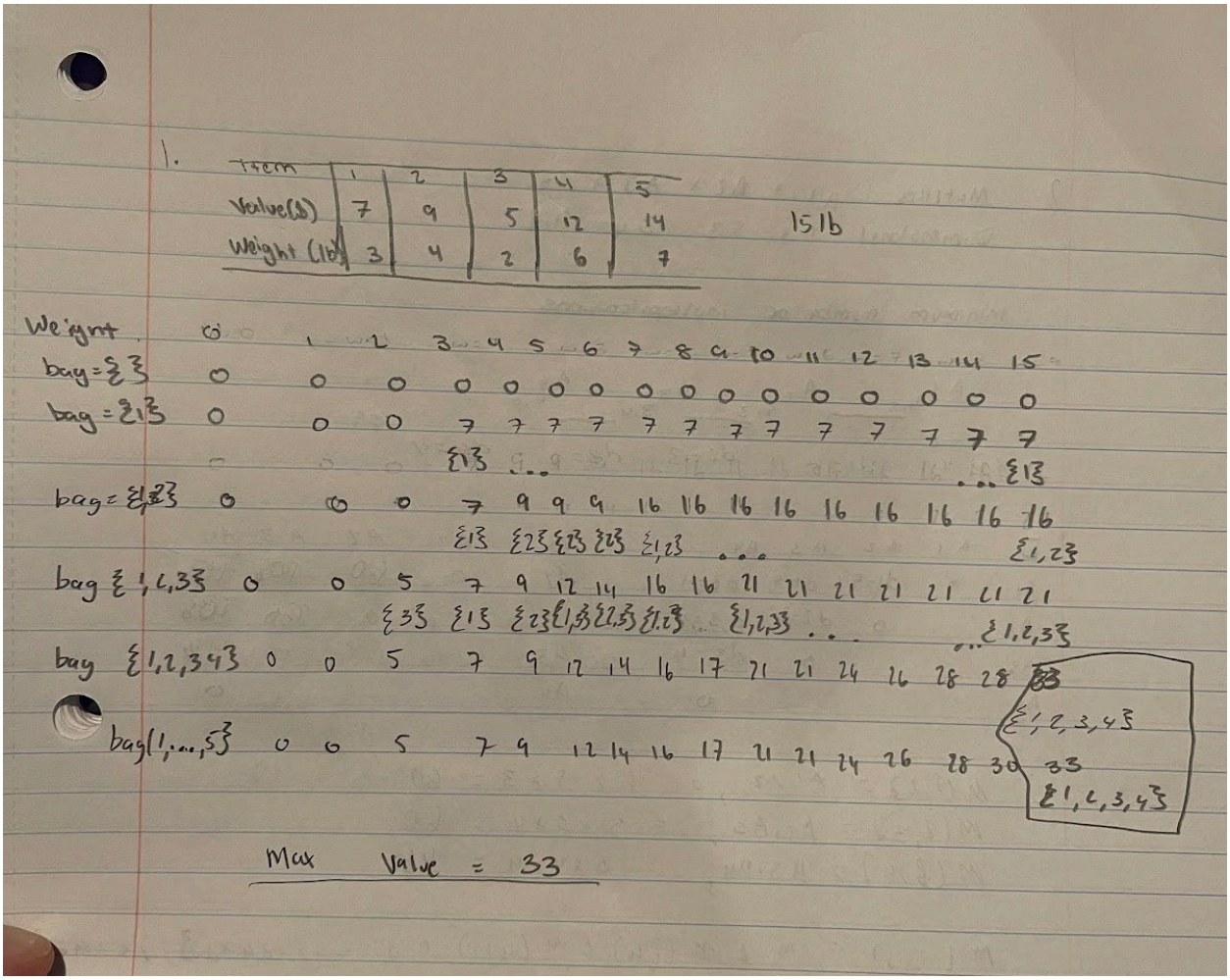
1. (15 points) Use the following table to find the maximum content of the sack using **Dynamic Programming**. The max weight the sack can hold is 15 lb. You May use the program to create the table.

item	1	2	3	4	5
Value($)	7	9	5	12	14
Weight(lb)	3	4	2	6	7

```
#include <iostream>
#include <iomanip>
#include <algorithm>

using namespace std;

// returns the maximum value that can be put in a knapsack of capacity W
int knapSack(int W, int wt[], int val[], int n) {
    int i, w;
    int K[n+1][W+1]; // K[n+1][W+1]

    // build table K[][] in bottom up manner
    for (i = 0; i <= n; i++) {
        for (w = 0; w <= W; w++) {
            if (i == 0 || w == 0) K[i][w] = 0;
            else if (wt[i-1] <= w) K[i][w] = max(val[i-1] + K[i-1][w - wt[i-1]],
K[i-1][w]);
            else K[i][w] = K[i - 1][w];
        }
    }

    for (int i = 0; i <= n; ++i) {
        for (int w = 0; w <= W; w++) cout << setw(5) << K[i][w];
        cout << endl;
    }
    return K[n][W];
}

int main() {
    // number of items and capacity of the knapsack
    int n = 5, W = 15;
    int val[5] = { 7, 9, 5, 12, 14 }, wt[5] = { 3, 4, 2, 6, 7 };

    int maxValue = knapSack(W, wt, val, n);
```

```
    cout << "\nMaximum value in the Knapsack = " << maxValue << endl;

    return 0;
}
```

WRITTEN WORK:



2. (15 points) Given the following chain of matrices multiplications

Matrices:	A1	*	A2	*	A3	*	A4
Dimensions:	4.5		5.3		3.4		4.4

Use Dynamic Programming to determine in what order we should multiply those matrices to obtain minimum number of multiplications. Show all work for maximum credit.

2. Matrices $A1 \times A2 \times A3 \times A4$

Dimensions: 4,5 5,3 3,4 4,4

Minimum number of multiplications

$A1_{4,5}$ $A2_{5,3}$ $A3_{3,4}$ $A4_{4,4}$

$d0 = 4$ $d1 = 5$ $d2 = 3$ $d3 = 4$ $d4 = 4$

D	A1	A2	A3	A4
A1	0	d1=5	d1=5	d3=4
A2		0	d2=3	d3=4
A3			0	d3=4
A4				0

M	A1	A2	A3	A4
A1	0	60	108	156
A2		0	60	108
A3			0	48
A4				0

$M(1,2) = A1 \cdot A2 = 4 \times 5 \times 3 = 60$ $M(1,1)$

$M(2,3) = A2 \cdot A3 = 5 \times 3 \times 4 = 60$

$M(3,4) = A3 \cdot A4 = 3 \times 4 \times 4 = 48$

$M(i,j) = \min \{ M(i,k) + M(k+1) + di-1 \times dk \times dj \}, i <= k < j$

$M(1,3) = \min \{ M(1,1) + M(2,3) + d0 \times d1 \times d3, M(1,2) + M(3,3) + d0 \times d2 \times d3 \}$

$i=1 \quad j=3 \quad k=1$

$= \min \{ 0 + 60 + 4 \times 5 \times 4, 60 + 0 + 4 \times 3 \times 4 \}$

$= \min \{ 140, 108 \} = 108$

$M[1,4] = 156$ min num of mul.

$M(2,4) = \min \{ M[2,2] + M(3,4) + d1 \times d2 \times d4, M(2,3) + M(4,4) + d1 \times d3 \times d4 \}$

$= \min \{ 0 + 48 + 5 \times 3 \times 4, 60 + 0 + 5 \times 4 \times 4 \}$

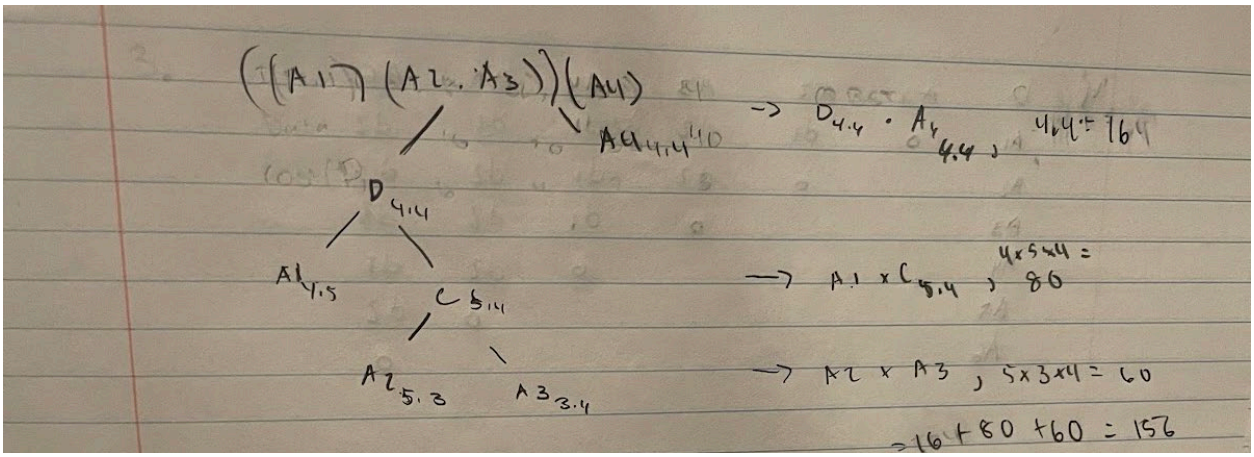
$= \min \{ 108, 140 \}$

$= 108$

$M(1,4) = \min \{ m(1,1) + M(2,4) + d0 \times d1 \times d4, M(1,2) + M(3,4) + d0 \times d2 \times d4 \}$

$= \min \{ 0 + 108 + 4 \times 5 \times 4, 60 + 48 + 4 \times 3 \times 4 \}$

$= \min \{ 188, 156 \} = 156$



3. (15 points) Use the following table to construct an OBST

Items	1	2	3	4
Data	10	20	30	40
Cost($)	6	4	2	3

Code to find optimal BST:

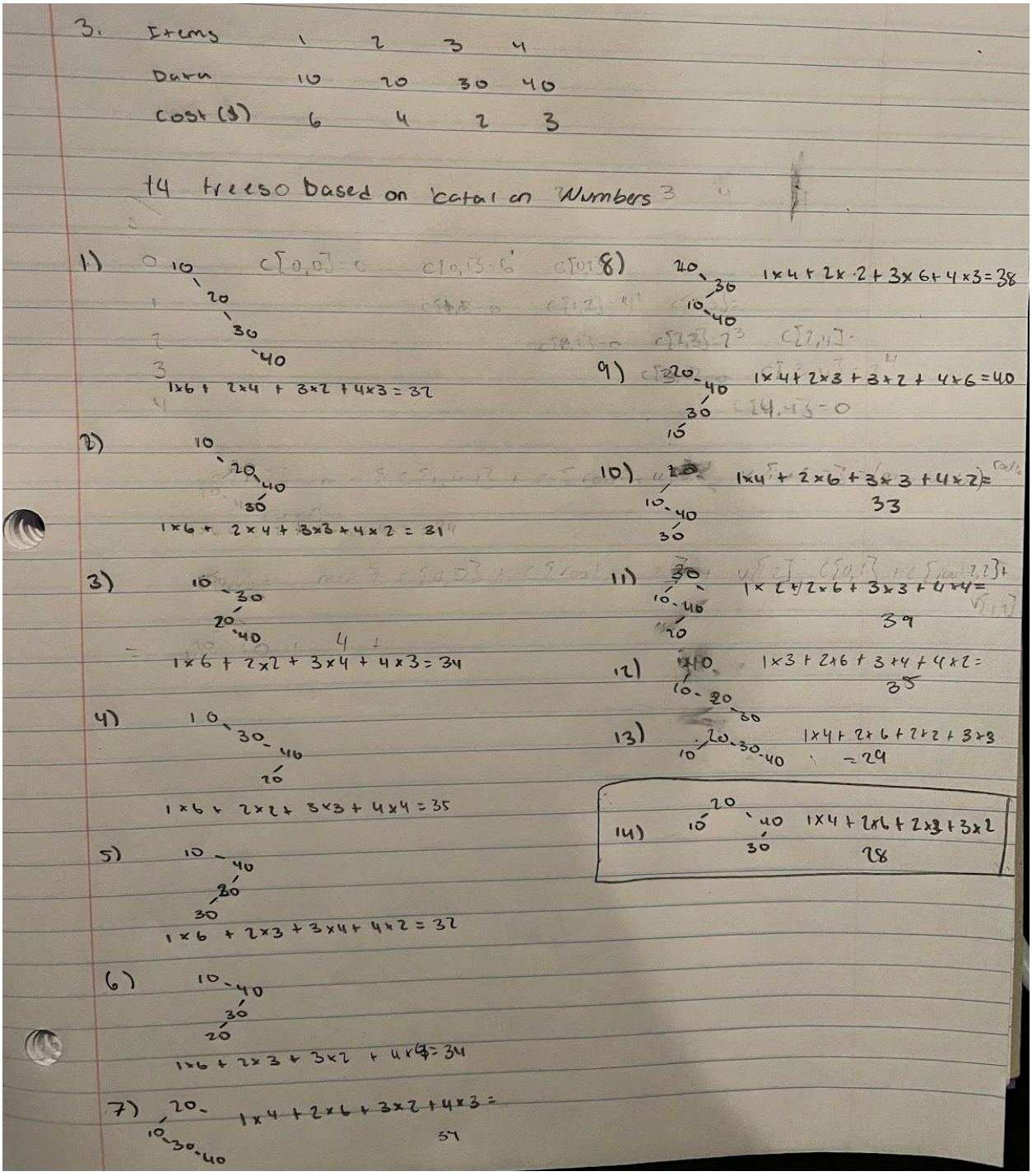
```
#include <bits/stdc++.h>
using namespace std;


int sum(int freq[], int i, int j);


int optCost(int freq[], int i, int j)
{
    if (j < i)
        return 0;
    if (j == i)
        return freq[i];

    int fsum = sum(freq, i, j);

    int min = INT_MAX;

    for (int r = i; r <= j; ++r)
    {
        int cost = optCost(freq, i, r - 1) +
                   optCost(freq, r + 1, j);
        if (cost < min)
            min = cost;
    }
    // Return minimum value
    return min + fsum;
```

```
}

int optimalSearchTree(int keys[],
                      int freq[], int n)
{
    return optCost(freq, 0, n - 1);
}


int sum(int freq[], int i, int j)
{
    int s = 0;
    for (int k = i; k <= j; k++)
    s += freq[k];
    return s;
}

// Driver Code
int main()
{
    int keys[] = {10, 20, 30, 40};
    int freq[] = {6, 4, 2, 3};
    int n = sizeof(keys) / sizeof(keys[0]);
    cout << "Cost of Optimal BST is "
         << optimalSearchTree(keys, freq, n);
    return 0;
}
```

OUTPUT:

```
[Running] cd "c:\Users\Nick\Desktop\Algorithm-Engineering-Work\Homework 4\" &&
g++ OBST.cpp -o OBST && "c:\Users\Nick\Desktop\Algorithm-Engineering-
Work\Homework 4\"OBST
Cost of Optimal BST is 28
[Done] exited with code=0 in 0.994 seconds
```

Written Work:



4. (10 points)Use the following table to group each chain of matrices to minimize the number of multiplications

D	A1	A2	A3	A4	A5	A6
A1	0	d1	d2	d1	d1	d2
A2		0	d2	d1	d2	d3
A3			0	d1	d2	d3
A4				0	d2	d1
A5					0	d2
A6						0

Complete the following table

Chain	Grouping with min multiplications (just write the grouping order)
A1.A2.A3.A4.A5.A6	
A2.A3.A4.A5.A6.A7	
A3.A4.A5.A6	
A4.A5.A6	

4.

D	A1	A2	A3	A4	A5	A6
A1	0	d1	d2	d1	d1	d2
A2		0	d2	d1	d2	d3
A3			0	d1	d2	d3
A4				0	d2	d1
A5					0	d2
A6						0

Chain

A1 A2 A3 A4 A5 A6

A2 A3 A4 A5 A6

(A7 DNE)

A3 A4 A5 A6

A4 A5 A6

Group

$= (A1\ A2), (A3\ A4\ A5\ A6)$, $D(A1, A6) = d2$

$= (A1\ A2)((A3\ A4\ A5)(A6))$, $D(A3, A6) = d3$

$= \boxed{(A1\ A2)(((A3\ A4)(A5)) \cdot A6)}$, $D(A3, A5) = d2$

$= (A2\ A3\ A4)(A5\ A6)$, $D(A2, A6) = d3$

$= \boxed{((A2) \cdot (A3\ A4))(A5\ A6)}$, $D(A2, A4) = d1$

$= (A3\ A4\ A5)(A6)$, $D(A3, A6) = d3$

$= \boxed{((A3\ A4) \cdot (A5))(A6)}$, $D(A3, A5) = d2$

$= \boxed{(A4) \cdot (A5 \cdot A6)}$, $D(A4, A6) = d1$

5. Write a program to display the first 10 Fibonacci numbers using the Cassini's identity $F(n-1)F(n-2) - [F(n)]^2 = (-1)^n$, n>=1

```cpp
#include <iostream>

using namespace std;

size_t s[100] = { 0 };

int cassini(int n) { return (n & 1) ? -1 : 1; }

size_t fib(const int& n) {
    if (s[n]) return s[n];
    if (n <= 2) return s[n] = 1;
    else
    {
        s[n] = fib(n-1) + fib(n-2);
        return s[n];
    }
}

int main(int argc, char** argv) {

    for(int i = 0; i < 11; i++){
        cout << fib(i) << "          " << cassini(i) << endl;
    }
    return 0;
}
```

OUTPUT:

```
[Running] cd "c:\Users\Nick\Desktop\Algorithm-Engineering-Work\Homework 4\" &&
g++ fib.cpp -o fib && "c:\Users\Nick\Desktop\Algorithm-Engineering-Work\Homework
4\"fib
1          1
1          -1
1          1
2          -1
3          1
5          -1
8          1
13          -1
21          1
34          -1
55          1

[Done] exited with code=0 in 0.421 seconds
```