-알고리즘 과제04-

과목명	알고리즘
교수명	최지웅
학 과	컴퓨터학부
학 번	20203123
이 름	이중곤
제출일	2023.12.07

목차

1.1 Algorithm 5.7 The Backtracking Algorithm for the 0-1 Knapsack Problem 구현 (Java)

```
public class Knapsack {

  private static final int N = 4; // number of items
  private static final int W = 16; // max weight of knapsack
  // 교재 입력 데이터
  private static final int[] w1 = {2, 5, 10, 5}; // weight of each item
  private static final int[] p1 = {40, 30, 50, 10}; // profit of each item
  // 자작 데이터
  private static final int[] w2 = {3, 4, 5, 6};
  private static final int[] p2 = {12, 12, 10, 6};
  private static final String[] include = new String[N];
  private static final String[] bestset = new String[N];
  private static int maxprofit = 0; // max profit
  private static int numbest = 0; // number of items in best solution

  public static void main(String[] args) {
    knapsack(-1, 0, 0);

    System.out.printf("maxprofit: %d\n", maxprofit);
    System.out.printf("bestset: ");
    for (int i = 0; i < numbest; i++) {
      System.out.printf("%s ", bestset[i]);
    }
  }

  private static void knapsack(int i, int profit, int weight) {
    if (weight <= W && profit > maxprofit) {
      maxprofit = profit;
      numbest = i + 1;
      for (int j = 0; j <= i; j++) {
        bestset[j] = include[j];
      }
    }

    if (promising(i, profit, weight)) {
      include[i + 1] = "YES";
      knapsack(i + 1, profit + p1[i + 1], weight + w1[i + 1]);
      include[i + 1] = "NO";
      knapsack(i + 1, profit, weight);
    }
  }

  private static boolean promising(int i, int profit, int weight) {
    int j, k;
    int totweight;
    double bound;

    if (weight >= W) {
      return false;
    } else {
      j = i + 1;
      bound = profit;
      totweight = weight;
      while (j < N && totweight + w1[j] <= W) {
```

```
            totweight += w1[j];
            bound += p1[j];
            j++;
        }

        k = j;
        if (k < N) {
            bound += (W - totweight) * p1[k] / w1[k];
        }

        return bound > maxprofit;
    }
  }
}
```

2. 교재 입력 데이터 테스트

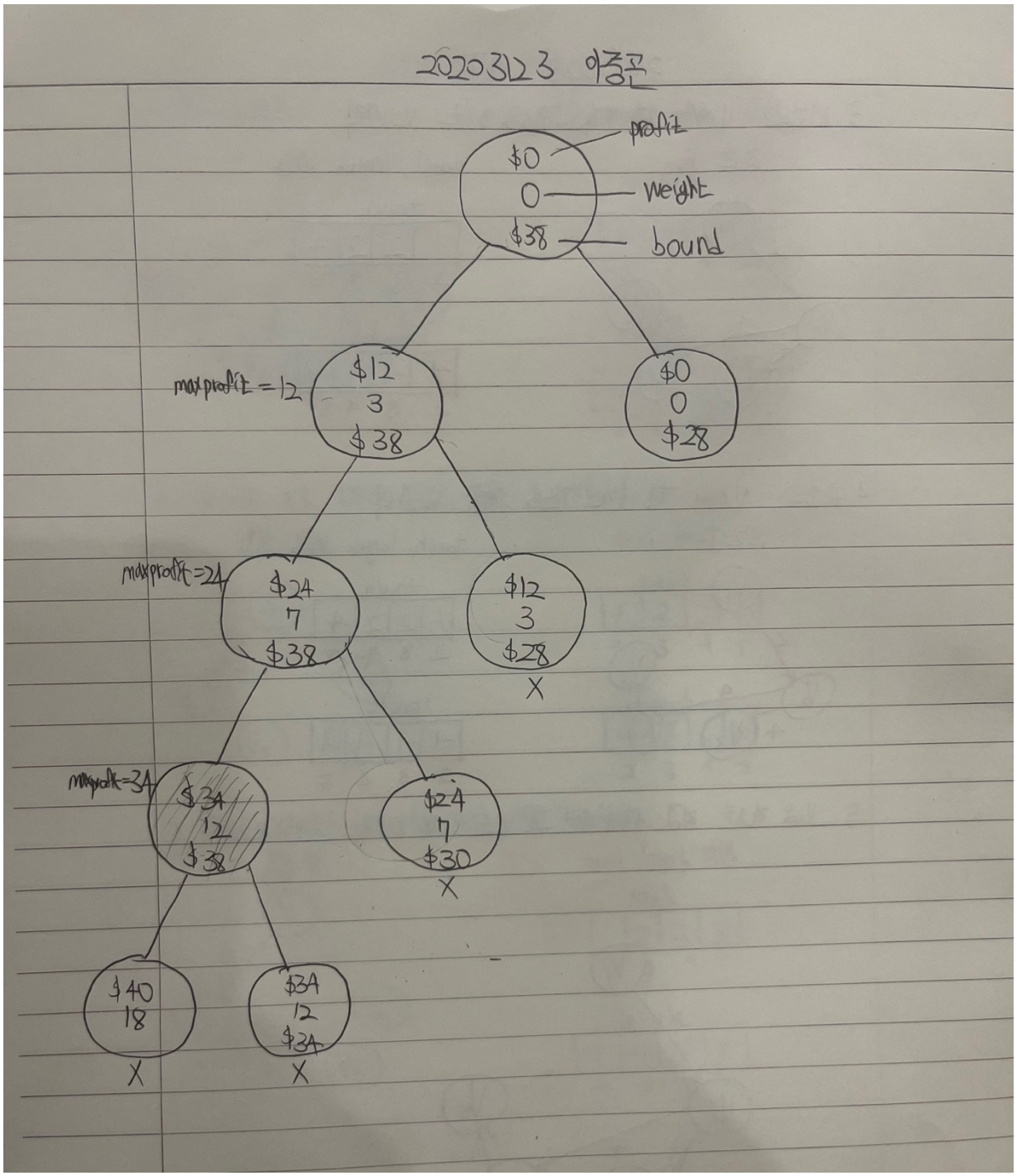
```
maxprofit: 90
bestset: YES NO YES
```

3. 자작 입력 데이터 생성 (Java)

```
// 자작 데이터
private static final int[] w2 = {3, 4, 5, 6};
private static final int[] p2 = {12, 12, 10, 6};
```

I	P2i	W2i	P2i/W2i
1	$12	3	$4
2	$12	4	$3
3	$10	5	$2
4	$6	6	$1

4. 자작 데이터를 이용한 알고리즘 손계산



5. 자작 데이터 테스트

```
maxprofit: 34
bestset: YES YES YES
```